Expert-driven approximation of MPE

Subtitle: Reinventing the World

Master's Thesis submitted to the
Faculty of Informatics of the *Università della Svizzera Italiana*
in partial fulfillment of the requirements for the degree of
Master of Science in Informatics
Artificial Intelligence

presented by
Thomas Francesco Tiotto

under the supervision of
Prof. Alessandro Facchini
co-supervised by
Prof. Alessandro Antonucci

September 2019

I certify that except where due acknowledgement has been given, the work presented in this thesis is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; and the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program.

Thomas Francesco Tiotto
Lugano, xx September 2019

To my beloved

Someone said . . .

Someone

Abstract

This is a very abstract abstract.

Acknowledgements

hello

Contents

Figures

Tables

Chapter 1

Introduction

1.1 Context

state the general topic and give background of what your reader needs to know to understand the problem outline the current situation evaluate the current situation (advantages/ disadvantages) and identify the gap

In genere mancano referenze

While neural networks and Artificial Intelligence (AI) - as a field - have existed for nearly seventy years, the concept of artificial intelligence dates back to at least Ancient Greece. In ancient times, artificial intelligence embodied in mechanical men was part of the domain of myth; in the twentieth century, of that of science. During this last decade, Artificial Intelligence can't anymore be described by a limited set of terms, as it has materialised out of Man's imagination, broken out of laboratories and has been given lease to act in the world at large.

What do you mean? Why "reductive"?

No sector of our economy has been left untouched by the recent and rapid rise of machine learning that has been enabled by the rediscovery of deep neural networks, the availability of Big Data and cheap parallel computing power. Fields as diverse and as critical as are government, healthcare, finance and bioinformatics have been revolutionised and the possibility has been set for new ones - such as self-driving vehicles - to be born. The ever increasing reliance of our society on ever more complex machine learning-driven algorithms can only make us worry ever more about the ethical dilemmas posed by such a situation. Our society has only very recently been confronted with the dilemma of assigning blame when a driverless car causes the death of a person but this moral problem is only the tip of the iceberg, even when focusing only on the automotive industry. For example, how should a self-driving car behave when confronted with a real-world analogous of the classic Trolley Problem - a situation where each course of action is liable to cause harm? On what basis should a person be denied a mortgage, access to university or a job interview? How can we be sure that there is no bias in the system? How do we even define if the system is behaving morally? Would it currently be feasible for a person that feels they have been harmed by such a decision to appeal it, as prescribed by the recent EU General Data Protection Regulation (GDPR)? As more and more decisions are made in an automated way, with many of them significantly impacting both individuals and society at large, it comes natural to stop and wonder what are the characteristics we would want the systems making these decisions to have.

Explainable AI (xAI) is the sub-field of AI that rests at the intersection between Computer

Science, Social Sciences and Philosophy and whose aim is to define our desiderata of artificially intelligent systems and machine learning algorithms from the point of view of their explainability. The basic idea is that the prerequisite for the evaluation of the ethical and moral implications of a machine's decision is for the system to be "interpretable" or "explainable". Within the xAI community, there is currently no unanimously agreed upon definition of which these desiderata should be or of the best way to implement them in real systems. There is also no common, agreed-upon, definition of what is meant by the phrase "understanding a system": some authors equate it to having a *functional understanding*, void of the low-level details, while others decline it into the concepts of *interpretation* and *explanation*, the former indicating the output of a format that a human user can comprehend and the latter a set of features that have contributed to generating the system's decision.

The difficulties start even in trying to define what interpretability really is. Does it mean to gain the trust the system's user? Of type of user in particular? Does trust stem from some property of the decisions the system makes or from some other inherent characteristic of the machine? A common approach to solving the difficulty in defining interpretability is to try and define it post-hoc by categorising systems into ontologies, based on their perceived interpretablility; unfortunately this seems like a circular way of approaching the problem: the classification of system models is being done utilising the same criterion that is trying to be uncovered by doing so. For reference, a commonly used classification is the following:

- **Opaque systems**: these are systems that offer no insight into the mapping between inputs and outputs; all closed-source algorithms fall under this definition;

- **Interpretable systems**: this is the vastest category, as the characteristic of these systems is *transparency* i.e. their inner workings are accessible but the onus of comprehensibility falls completely onto the user. The classical example is that of neural networks where the mapping from inputs to outputs (the *weights*) is inspectable by the user who can, theoretically and depending on her skill, interpret them;

- **Comprehensible systems**: systems falling into this category emit additional symbols together with their outputs with the explicit intent of giving the user the means to interpret and understand the automated decisions; the additional symbols may be visualisations, natural-language text or any other means of demystifying the output. These extra symbols would need to be graded based on the user's expertise, as comprehension is a property that involves both man and machine but materialises on the human side.

Some authors propose to classify systems as *non-interpretable*, *ante-hoc interpretable/transparent* and *post-hoc interpretable*; this roughly corresponds to the ontology presented above.

What I hope can be gleamed from this brief introduction to the field of Explainable Artificial Intelligence, is that many of the problems it aims to tackle are hard *per-se* and may not have a unique optimal solution. This is because these issues are not only engineering problems, but exist at the intersection between man and machine and as such can't be tackled using only the methods of Computer Science. There is no way to satisfactorily investigate the human element of the situation without resorting to the well-established methods of the Social Sciences. There is little hope to know in which direction to procede without the guiding force that can only come from philosophy, because of its millennia-long tradition in thinking about ethical and high-level issues. It should be clear that when the human - and particularly the ethical - domain are part of the equation, it is impossible *by definition* to find an optimal and unique solution.

references to articles ?

explain better what follows here, since it is portant

what does functional mean?

what are low level details

??? explain better. gives examples, help th reader

does not sound well expressed here. also w has trust to do with interpretability? expla intuitions, motivations

why are you speaking about ontologies her not simply of categories etc?

in general. explain what a system, input a put are

References!!!

for instance? which methods? example?

you have to link what you just said with what you are gonna do later. for instance, illustrate what you are saying with examples. use one close to what you are going to do. also, you speak about NN but never about graphical methods, such as BN. you have to speak about them here too.

1.2 Problem and Significance

identify the importance of the proposed research - how does it address the gap? state the research problem/ questions state the research aims and/or research objectives state the hypotheses

again, references!

AI has a trust problem. The bigger problem with AI is not anymore its utility, as that has mostly been solved by deep neural networks, but its capacity to elicit the trust of the users. To be truly useful, an automated system should be able to make itself be trusted in a manner proportional to the criticality of its application. Unfortunately, the explainability and, by extension, the "trustability" of machine learning models are inversely proportional. There are many examples of modern methods - such as boosted trees, random forests, bagged trees, kernelized-SVMs - that show this tendency, but it is best exemplified by *deep neural networks* (DNN). Deep Neural Networks are machine learning models constructed by stacking many layers of artificial neurons, these systems are currently state of the art on a variety of tasks but are among the least easily interpretable systems due to the fact that they represent information in an implicit and distributed manner among their network weights. Some older methods, like decision trees or rule-based methods, are inherently more interpretable due to their simplicity and the fact that they can explicit state their reasoning steps, but are less accurate and flexible than more modern techniques.

ok but come on, there is not only NN in th world

The runaway success obtained by modern Machine Learning in a variety of domains, on a spectrum that goes from engineering to social work, has created the desire to also start applying these methods to mission-critical and traditionally more entrenched fields. A perfect example of a field exhibiting both these characteristics is that of medicine. The first successful artificially intelligent systems date back to the 1970s and '80s and were based on *symbolic methods* integrated with *knowledge-bases*. These systems were by design capable of providing an explanation for their reasoning and were thus accepted by the medical community in an implementation known as *expert systems* that aimed to perform functions similar to those of a human expert. The deficiency of modern AI methods in being able to provide causal links for their reasoning process has held back their acceptance in the field of medicine, regardless of their superior performance and accuracy.

In a high-stakes domain such as the medical one, it would be unthinkable for a doctor to trust the predictions of an AI system a priori; any decision with profound moral implications - such as prescribing or interrupting the treatment of a patient - would have to first be validated by a human. The possibility of carrying out this validation and its quality are dependent on the degree of interpretability of the model that made the decision. Unfortunately, as has been repeated many times, the best performing models are often also the most opaque to inspection.

Explainability is not a necessary condition only for the verification of the system which, as we have just discussed, is a presupposition for it to be applied in mission-critical domains, but also for the extraction of knowledge from data. The amount of information that a machine learning model can process is many orders of magnitude greater than that inspectable by any human; this may let a computer spot new patterns in the data that aren't immediately apparent or are latent given only a moderate amount of samples. Being able to turn this information into

not clear. you were speaking about "valida before. is this taken as a synonymous of "v ification"? if yes, why two words for the sa concept? if not, then explain better

why?

new knowledge implies the system having the ability to output human-interpretable symbols that are capable of communicating it in a comprehensible and effective way.

so, what is knowledge? why explainability essary to create knowledge? you do not ex this here, imho

There has recently been much research carried out on trying to explain and extract knowledge from deep neural networks together with attempts to marry the connectionist and symbolic approaches to artificial intelligence - a subfield known as *neuro-symbolic computation* while also reconsidering mixed approaches such as *Bayesian Networks*. A Bayesian Network is a graphical and computationally efficient way of representing dependencies between random variables. The graphical component is immediate as in the model each random variable is represented by a node of a Directed Acyclic Graph (DAG), with the edges connecting them standing for their dependencies. The efficiency stems from the fact that the graph structure imposes a factorisation of the joint probability space and thus lets each variable be calculated using only the values of its parents.

again, a lot of nice talk about NN, but then little about BN and thus what you are gonna do

1.3 Response

outline the methodology used - outline the order of information in the thesis - a roadmap - Maximum 2500 words.

The work carried out in this thesis concentrates on explainability in the medical domain and presents both a practical part, with the implementation of a Bayesian network-based system focused on *knowledge-extraction*, as defined at the end of the previous section, and a theoretical one, regarding the definition and validation of desiderata for an artificially intelligent system using the aforementioned system.

The implemented system aims at supporting medical decision making through the instauration of a dialogue with the user/domain expert. To this end, the information implicit in the data is used as basis for a constructive dialogue with the user; this starts with the expert informing the system of which knowledge is certain i.e. a variable's value that has been observed in a specific patient, and continues via a process where the next most probable *(variable, state)* pair is proposed, with the expert having the choice of accepting it or refusing it, if she believes that the variable under examination doesn't adequately explain the accumulated evidence. Each accepted variable is added to the evidence set, as the system gives priority to the domain expert's judgement. The result of the dialogue is an *explanation tree* whose nodes represent *(variable, state)* pairs and are organised into branches, depending on the flow of the dialogue; more specifically, there will always be a *main branch* corresponding to the choices of the user and none or more *alternative branches* whose role is to inform the expert of the possible alternative outcomes to his decisions.

This software system was developed and tested tested in collaboration with *Istituto Cantonale di Patologia*, a medical institute in Locarno, Ticino, Switzerland that specialises in the analysis of tissue samples received from hospitals, clinics and private doctors. Its main activity is to characterise the samples by using *histo-cytopathologic techniques*, with particular focus on the diagnosis of cancer and tumoural diseases in general.

The theoretical part of this thesis aims to understand how an

Chapter 2

Literature review

2.1 Introduction

Through a review of recent relevant literature, this chapter will clarify the concept of *explainability*, that is central to the field of Explainable AI. It will then move on to a discussion where a justification will be given for the importance assigned to explainability in our contemporary societies. The evaluations methods that have been proposed to measure explainability will next be assessed. The analysis will then focus on an assessment of the previous concepts as applied specifically to Bayesian Networks. Finally, the recent paper "Explaining the Most Probable Explanation" by Butz et al. [2018] will be reviewed and connected to the previously analysed notions.

Throughout the chapter, various gaps present in the literature will be identified and assessed; these will be summarised in a coherent fashion in Sec. 2.7.

2.2 Explainability

Doran et al. [2018], based on a frequency analysis of explanation terms within documents from relevant research communities, highlight how different circles have different approaches to the concept of explainability and how, even within the same group, terms are used interchangeably. In particular, they note the overloading of the notion of "explainability" with that of "interpretation"; a concept that is often defined within the xAI community as necessary for, but distinct from, explainability. The use of "interpretable" as signifying the property belonging to a system whose inner workings are accessible can be found, for example, in the recent paper by Gilpin et al. [2018]. In other recent works the two terms are conflated, for example by Mittelstadt et al. [2019], Guidotti et al. [2018] and in the influential work by Doshi-Velez and Kim [2017]. This seems to prove the point that Lipton [2016] makes in the widely-cited paper "The Mythos of Model Interpretability", that "the task of interpretation appears underspecified. Papers provide diverse and sometimes non-overlapping motivations for interpretability, and offer myriad notions of what attributes render models interpretable."

Most works, even those that blend the notions of interpretability and explainability, seem to agree on the end-goal that implementing such a concept should have; that is, to "summarize the reasons for neural network behavior, gain the trust of users, or produce insights about the

causes of their decisions" (Gilpin et al. [2018]) by being able to "explain or to present in understandable terms to a human" (Doshi-Velez and Kim [2017]). Where the consensus diverges, in defining what constitutes an explanation and the desiderata that it may have. Mittelstadt et al. [2019] identify, within the literature, two broad classes of interpretability/explainability: *transparency* (also called *ante-hoc* explainablility) and *post-hoc* interpretability. The former type deals with the internal workings of a system while the latter applies to its external behaviour. Lipton [2016] identifies three explanations that can make a model transparent: a mechanistic understanding of the workings of the system in its entirety, of the individual components or of the algorithm. A system may be made post-hoc interpretable by way of, among others, natural language explanations, visualisations or interactive interfaces. These methods often do not precisely clarify the exact working of a model, but "they may nonetheless confer useful information for practitioners and end-users of machine learning." Biran and McKeown [2017] notes how the transparent or *white-box* paradigm was sufficient for classic rule-based models but - with the advent of contemporary machine learning models - is no longer useful. They argue that it is nowadays unreasonable to expect that any domain expert be able to understand a prediction if they are not also a machine learning specialist. To address this issue, they propose a Natural Language Generation system; that is, a post-hoc explanation in Lipton [2016]'s categorisation.

A widely-recognised feeling, tightly connected with the already recognised lack of shared working definitions, seems to be that researches of explainable AI are ignoring the enormous corpus of existing models in philosophy, psychology, cognitive and social sciences and human-computer interaction. This feeling of disconnect is echoed by Gilpin et al. [2018] who points out how philosophical texts have long debated what constitutes an explanation, by Mittelstadt et al. [2019] who explicitly says how "many different people, be they lawyers, regulators, machine learning specialists, philosophers, or futurologists, are all prepared to agree on the importance of explainable AI [...] very few stop to check what they are agreeing to, and to find out what explainable AI means to other people involved in the discussion" The fact that explainable AI researchers seem to be intent on "reinventing the wheel" is stated most strongly by Miller [2018], whose paper "Explanation in Artificial Intelligence: Insights from the Social Sciences" is based on the premiss that "most of the research and practice in this area seems to use the researchers' intuitions of what constitutes a 'good' explanation" and argues for the adoption of the existing research in the social sciences. The author's views are well summarised by the position xAI is set to occupy in Fig. 2.1. The feeling is that explainable AI researchers are terming their methods "explanation" based on purely personal views and are thus building explanations that only work for themselves; in other words, "the inmates are running the asylum" (Miller et al. [2017]). The following quote by Guidotti et al. [2018], in my view, perfectly sums up the state of the research in the field: "It is evident that the research activity in this field completely ignored the importance of studying a general and standard formalism for defining an explanation, identifying which are the properties that an explanation should guarantee, e.g., soundness, completeness, compactness and comprehensibility. Concerning this last property, there is no work that seriously addresses the problem of quantifying the grade of comprehensibility of an explanation for humans, although it is of fundamental importance."

To remedy to this state of affairs, there have been a number of works, such as Doshi-Velez and Kim [2017]'s, that attempted to define what an explanation means and to reach consensus on it. The most compelling attempt is in the paper "What Does Explainable AI Really Mean? A New Conceptualization of Perspectives" where Doran et al. [2018] try to synthesise the current state of affairs into a taxonomy of models:

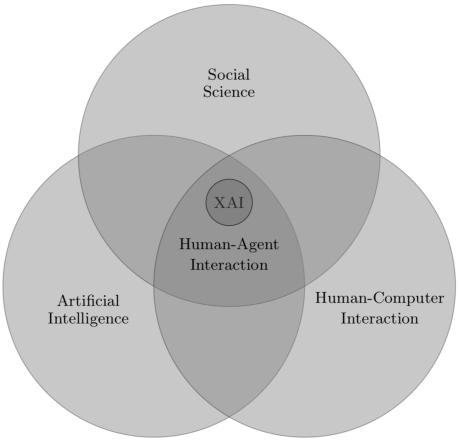


Figure 2.1. Miller [2018]

- Opaque systems: systems where the mapping from inputs to outputs is entirely inaccessible to the user.
- Interpretable systems: systems whose inner workings are inspectable from the outside, but that makes no effort to clarify them.
- Comprehensible systems: systems that emit extra information together with their output.
- Explainable systems: systems that explicitly output a human-understandable line of reasoning i.e., an explanation.

It is recognised, thus implicitly accepting the view that xAI should learn from the social sciences, that comprensibility depends not only on the system's characteristics, but also on the user's ability and knowledge. Comprensibility and Interpretability are seen as separate concepts as comprehension requires transparency but interpretation does not, as the user may reason over only the emitted extra symbols. This notion of comprensibility is expanded into that of *real* explainability, that is based on a notion of "ability to formulate, for the user, a line of *reasoning* that explains the decision making process of a model using *human-understandable features of the input data*." (italics by the authors).

2.3 Importance of Explainability

As noted by Edwards and Veale [2018], "businesses and governments are increasingly deploying machine learning (ML) systems to make and support decisions that have a crucial impact on everyday life" so, as Gilpin et al. [2018] say, "it becomes necessary for these mechanisms to explain themselves". This feeling of urgency and purpose is echoed throughout the reviewed literature; it seems, that even if researchers and the field as a whole cannot agree on a definition of explainability (as discussed in Sec. 2.2), there is a keen awareness on the need for models to be explainable. This intense urge to define explainability and, at the same time, try and create models exhibiting this property, may be counterproductive as the field risks fragmenting into a series of diverging strands, as noted by Abdul et al. [2018] and visualised in Fig. 2.2.

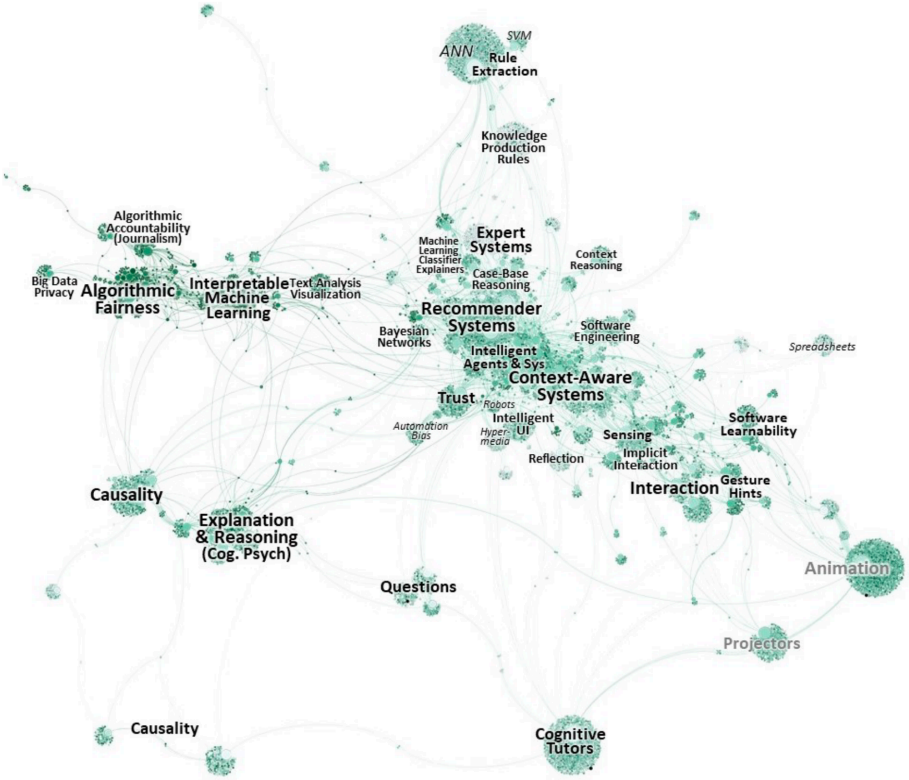


Figure 1. Citation network of 12,412 papers citing 289 core papers on explanations. This shows several communities of research domains, how they are closely related due to co-citations by citing papers and trends of research interest on explainable systems. Each node (circle) indicates a paper. Node size (of core papers) indicates the logarithm of the number of citing papers. Node color indicates the paper's age; darker green for more recently published papers. The nodes and edges are clustered by a force-directed layout method (ForceAtlas2 in Gephi [83]). Nodes are closer if more citing papers co-cite the same core papers. For example, recommender systems, intelligent agents and context-aware systems are closely related in terms of explainable systems, but the recent work on algorithmic fairness and interpretable classifiers are mostly developing independently. The size of the text roughly corresponds to the number of papers identified.

Figure 2.2. Abdul et al. [2018]

There are a myriad of reasons that are brought forth as a justification for the development of explainable AI, and we will review these in the following paragraphs, but it seems timely to start with one in particular: the need introduced by the European Union's broad General Data Protection Regulation (GDPR). The GDPR was approved by the European Parliament in 2016 and came into effect in 2018. More than one author cites an urgency to conform to this regulation (for example Doshi-Velez and Kim [2017], Gilpin et al. [2018]), most likely referring to Article 22 of the regulation that, supposedly, mandates for a "right to an explanation" of algorithms. While algorithmic explainability is undoubtedly a commendable end-goal, it may be the case that this reason, in particular, to strive for it be a false one. Edwards and Veale [2018] posit that Article 22 of the GDPR actually does not contain the publicised right to an explanation but is "merely a right to stop processing unless a human is introduced to review the decision on challenge" and as the authors point out, there are, nowadays at least, very few systems without a human in the loop. Secondly, there is no mandate for the "explanation" to be human-understandable, so the obtained result may actually be no explanation at all. If the analysis of Edwards and Veale [2018] were correct, then the urgency advocated by many researchers on the grounds of conforming to the GDPR would turn out to be based on no valid reason.

A second motive brought forth for the necessity of explainability, for example by Gilpin et al. [2018] and Abdul et al. [2018], is that comprehensible models are much more likely, or even necessary, to engender users' trust. While this may very probably be the case, no motives are given for why this should be the purpose of explainable AI and not just a desirable by-product of obtaining explainability.

Other authors, for example, Doshi-Velez and Kim [2017] and Guidotti et al. [2018], frame the issue as one of moral necessity. One need not look far to find examples of ML models displaying covert bias or making decisions we would regard as unethical; a more in-depth investigation would reveal that this was the case even before the popularisation of "black box" models as the Deep Neural Networks. Guidotti et al. [2018] give a reasonably comprehensive list of classic cases that show the risks of not having comprehensible AI. The oldest of these dates back to the 1970s and 1980s and tells of a system used to screen job applicants that, even though programmed to ignore people's ethnicity, was still seen to discriminate against minorities. In the same vein but much more recent, the American Military discovered that their computer vision system, that was developed to differentiate between enemy and friendly tanks automatically, had poor accuracy because it had learned to use the background information of the test set photos instead of the pixels representing the actual tanks. Both these cases exemplify how an algorithm may make "wrong" inferences based on spurious or latent information that was already in the data set but that no human could have imagined being relevant. Other failures epitomise how a model may learn our own social biases; for example, a recent Princeton study (Caliskan et al. [2017]) proved how models trained on web text corpora showed marked biases (towards race, gender ...) that reflected our the ones present in our own society. Because of their findings, the authors went as far as to suggest that transparency would not be enough to uproot biases, as the very *semantics* reflects biases latent in our culture.

The driving motivation and sense of urgency present in all of the reviewed literature is quite certainly tied to the renewed interest and applicability of AI. Preece [2018] claim that the interest for explainability is naturally linked to the interest in AI itself; if this were true, then it would confirm that a need for transparency is implicit in the field itself. This would validate the assertion made by Doshi-Velez and Kim [2017], that the need for an explanation stems from an incompleteness in the formalisation. Lipton [2016] also acknowledges this and states that "the

demand for interpretability arises when there is a mismatch between the formal objectives of supervised learning and the real world costs in a deployment setting". In the presence of such uncertainty, rendering the resulting model open to inspection would make the "gaps in problem formalization visible to us" and thus enable us to apply our best human judgement to evaluate them and their consequences (Doshi-Velez and Kim [2017]).

2.4 Evaluation of Explainability

As concluded in Sec. 2.3, the fact that ML models operate on incomplete assumptions makes it a necessity to have some form of evaluation of their performance. As Lipton [2016] states, "it turns out that many situations arise when our real world objectives are difficult to encode as simple real-valued functions" and this could lead to evident difficulties in optimising with respect to soft, but of paramount importance, concepts such as ethics and legality. Being able to evaluate an automated explanation lets us "serve those objectives that we deem necessary but struggle to model formally".

Unfortunately, the finding outlined in Sec. 2.2 that there is no consensus on the definition of explainability also necessarily entails that there is no agreed-upon methodology to evaluate such a property. Doshi-Velez and Kim [2017] note as much when they comment "unfortunately, there is little consensus on what interpretability in machine learning is and how to evaluate it for benchmarking". This makes perfect sense because trying to evaluate something without first having defined it, is worse than trying to hit a moving target. Once again, the feeling among many authors is that "inmates are running the asylum".

Some authors have tried to put some order in the barrage of methods; Doshi-Velez and Kim [2017] provide one of the most compelling attempts. Doshi-Velez and Kim [2017] set out to outline a taxonomy, having noted a "lack of rigour" and how current interpretability approaches usually fall into two categories: interpretability in the context of an application and interpretability via a quantifiable proxy. The former approach assumes that "if the system is useful in either a practical application or a simplified version of it, then it must be somehow interpretable"; the latter sees researchers claim that a model class is interpretable and then present algorithms to optimise within that class. In their words, both classes rely on a notion of "you'll know it when you see it". The taxonomy the authors lay out is shown in Fig. 2.3 and borrows from methods already standard in human-computer interaction and visualisation; the guiding ideal is that "evaluation of applied work should demonstrate success in the application" and thus the best sort of evaluation is the one that involves humans the most. Thus, at the lowest level, they pose "Functionally-grounded Evaluations" that require no human in the loop and evaluate the quality of an explanation using some proxy measure; the advantage is the low cost, but the tradeoff is a lack of specificity. A proxy measure that has already been human-validated, for example a decision tree, a set of rules or a linear model (Guidotti et al. [2018]) as regards its explainability, may be an appropriate measure for more exotic systems. The second level of evaluation, "Human-grounded Evaluation", involves humans, albeit not domain expert ones; this kind of setup enables the testing of more general notions of explainability. Finally, "Application-grounded Evaluation" is taken as the gold standard to demonstrate success in the context of an application; the authors claim that there is no better way to evaluate the quality of an explanation that having a domain expert test it in the context of a real task. In their words, "the best way to show that the model works is to evaluate it with respect to the task": "for example, a visualization for correcting segmentations from microscopy data would be evaluated

via user studies on segmentation on the target image task; a homework-hint system is evaluated on whether the student achieves better post-test performance. Specifically, we evaluate the quality of an explanation in the context of its end-task, such as whether it results in better identification of errors, new facts, or less discrimination".

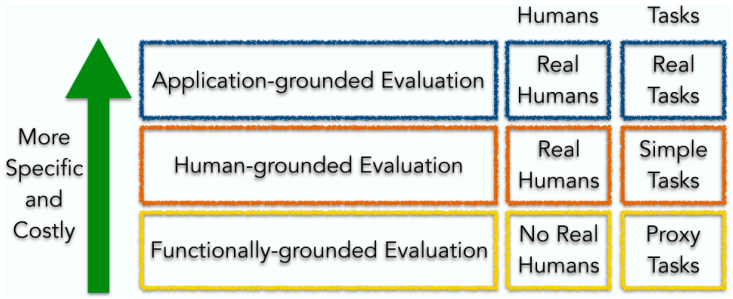


Figure 1: Taxonomy of evaluation approaches for interpretability

Figure 2.3. Doshi-Velez and Kim [2017]

Guidotti et al. [2018] outline a taxonomy developed along a different axis; specifically they identify: methods to explain black box models, methods to explain black box outcomes, methods to inspect black boxes and methods to design transparent boxes. An essential factor that the authors identify is that evaluation is a graded notion, as also noted by Gilpin et al. [2018]: different users, with different expertise and background, may rate the quality of explanations differently. Another point they make that is quite novel is that time should also be part of an evaluation; depending on the available time a user may prefer a more straightfoward or more elaborate explanation. They end by noting that very few works take user expertise and the time taken to understand the proposed explanation into account.

The neglect of the human side of explanations is also lamented by Abdul et al. [2018] who see researchers focusing on creating mathematically-explainable models at the expense of ones that are usable and practical in real-world situations. Again, the underlying issue seems to be that xAI researchers are unaware or ignoring the sizeable corpus of research available in cognitive psychology, human-computer interfaces and philosophy. The author sees the opportunity for HCI to bridge the gap between models and users by way of an interactive approach, as opposed to the mainstream static explanations being proposed in the literature. An interactive explanation may take the form of a dialogue or of various visualisation techniques; the defining characteristic of such a mode is that it lets the user freely explore the system's behaviour. Guidotti et al. [2018], while discussing the types of data used in ML models, lend credence to the goodness of this output modality with the statement that "other forms of data which are very common in daily human life are images and texts. They are perhaps for human brain even more easily understandable than tables".

Guidotti et al. [2018], though, take the view that evaluation of model *complexity* should be equated to its *comprehensibility*, that is not something that often appears in the relevant literature. Basically, the authors are advocating for the use of complexity as a proxy model for explainability, if we are framing the issue using the taxonomy proposed by Doshi-Velez and Kim [2017] (Fig. 2.3). This may very well be a valid approach, but there is no supporting evidence

for it in the paper itself.

A useful reference for how to set up an experiment falling into the class of either Human-grounded or Application-grounded Evaluation can be found in Stumpf et al. [2009]'s "Interacting meaningfully with machine learning systems: Three experiments" 2009 paper. In this work the authors set up "three experiments to understand the potential for rich interactions between users and machine learning systems"; the first, and most relevant, was a think-aloud study that investigated "how machine learning systems should explain themselves to end users, and what kinds of improvement feedback end users might give to the machine learning systems". These studies are interesting as a blueprint for future human-centred evaluations, of the type whose absence is being lamented by many authors.

Mittelstadt et al. [2019] summarise the existing critiques to offer a clear and direct evaluation of the field of Explainable AI as a whole, when they state that "no matter the approach taken in xAI, reflexivity *[taking account of itself or of the effect of the personality or presence of the researcher on what is being investigated], my note* is needed to ensure the community actually works towards its normative and practical goals to render models holistically transparent or provide high-quality post-hoc interpretations of model behaviour. Critical questions must be repeatedly asked and answered. For example, will the methods developed make machine learning models more interpretable? More trustworthy to users? More accountable? And to whom will explanations be accessible, comprehensible, and useful? Answering such questions requires considering the methods developed in xAI in the context of prior work in fields addressing such normative and social questions. Local and approximation models may in fact resemble existing, well-known approaches to explanations in the 'explanation sciences', which would provide insight". They then conclude by stating that "xAI generally avoids the challenges of testing and validating approximation models, or fully characterising their domain". From the review of the current state-of-the-art carried out in this section and Sec. 2.2 and 2.3, these could both be seen as entirely valid criticisms. It really seems that the field of xAI as a whole should try and reposition itself as suggested in Fig. 2.1 and not try to build methods from first principles, many of which may be outside the domain of expertise of the researchers in question.

2.5 Explainability of Bayesian Networks

Bayesian Networks have enjoyed widespread appeal as a Machine Learning method, especially inside the medical domain. This may be because the formalism (introduced in detail in Sec. 3.5) "offers a natural way to represent the uncertainties involved in medicine when dealing with diagnosis, treatment selection, planning, and prediction of prognosis. This is due to the fact that the influences and probabilistic interactions among variables can be described readily in a BN" (Lucas [2001]); that is, even if the BN model is *complete*, in the sense that every possible probabilistic statement can be computed in it, it also is easy to combine multiple variables of interest into composite questions. Another attractive feature of BNs is their relatedness to the class of *Causal Networks* that were popularised by the groundbreaking work of Pearl and Dechter [1988]; for all intents and purposes, a Causal Network is simply a Bayesian Network where all the relationships represent a causal effect. Nonetheless, BNs are not considered as inherently interpretable by the literature, and thus, a series of methods were developed to address this shortcoming. Timmer et al. [2015] note this in the introduction to their paper by stating that "for non-statistical experts, however, Bayesian networks may be hard to interpret. Especially since the inner workings of Bayesian networks are complicated they may appear as black box

models", "the interpretation of BNs is a difficult task, especially for domain experts who are not trained in probabilistic reasoning".

The best overview of the state of explainability in BNs is given by Lacave and Díez [2002] in the paper "A review of explanation methods for Bayesian networks"; here the authors identify various classification criteria for an explanation given by a BN:

- *description* vs. *comprehension*: the former consists in displaying the data set or providing further details regarding the output, the latter attempts to guide the user in understanding the model's conclusions.

- *micro-level* vs. *macro-level*: detailed description of how a single node is affected vs. showing the main lines of reasoning.

- *verbal* vs. *graphical*: "the most direct and intuitive way of showing the information embodied in a Bayesian network is to display the corresponding graph". When presenting probabilities, Henrion and Druzdzel [1990] strongly suggest that these be "linguistic probabilities" i.e., for the quantitative probabilities inherent in the model to be converted to a qualitative equivalent. This is validated by research showing that linguistic expressions of probability are better understood than the equivalent numerical representation. Some of the many models surveyed in the paper user colours, shading and line thickness to represent the salience of links and nodes.

The authors also identify the three components of BNs that need to be explained: the knowledge base, the reasoning process and the evidence propagated. The first of these "consists of determining which values of the unobserved variables justify the available evidence" and is, in general, done by finding the solution to the Most Probable Explanation problem (defined in Subsec. 3.5.2). The explanation of the model is considered a static explanation (as opposed to dynamic ones, that will shortly be covered) and is simply the process of verbally or graphically displaying the information already present in the data. The final element to explain, that is what would most commonly be called an explanation in xAI circles, is the reasoning behind the model's outputs; a system may accomplish this by providing a justification for its outputs, for the results it did not give or via hypothetical reasoning. The first of these is maybe the most important, because it is paramount for any system, not just a BN, to be able to explain the reasons behind its outputs; returning to the medical setting, it was seen that physicians, in particular, are very reluctant to accept the advice of a machine if they can not understand how it was obtained. A BN, unlike other ML systems, can also innately exhibit evidence for why it did not provide the output expected by the user and can also reason *counterfactually* i.e. provide alternative outputs. These last two capabilities are particularly important, from an explainability perspective, in light of Miller [2018]'s findings regarding the nature of explanations.

Miller [2018]'s paper "Explanation in Artificial Intelligence: Insights from the Social Sciences" investigates what constitutes an explanation from a psychological perspective; the conclusions are that explanations possess four primary characteristics:

- explanations are *contrastive*; that is people do not ask why an event happened by why another event did not happen instead. A Bayesian Network, as noted in the previous paragraph, is capable of modelling counterfactuals which enables them to give contrastive reasons naturally.

- explanations are *selected*; people expect an explanation to be selected based on some

cognitive bias; they do not expect a complete recount of all causes of an event. A BN has the ability to combine an output from its constituent variables in a flexible manner.

- to people, probabilities are not as important and not as well understood as causal relationships. BNs, as already mentioned, are closely related to *graphical causal models*, so their explanations have the possibility of being based on causal grounds (Lipton [2016], Rani et al. [2006]).

- explanations are *social*; that is they involve an *explainer* and an *explainee*. This is recognised in the most important work on the social aspects of conversation, Hilton [1990]'s "Conversational processes and causal explanation", that supports the view that an *explanation* is a *conversation*. A dialogue is an example of a *dynamical explanation*, in the framework set out by Lacave and Díez [2002]; these authors also recognise that an explanation "always means explaining something to somebody" and that thus "one of the key features of an effective explanation is the ability to address each user's specific needs and expectations, which primarily depends on the knowledge he/she has". So "In the case of a Bayesian network, the explanation generated for a user that is familiar with the concepts of prevalence, prior/posterior odds and likelihood ratios should be very different from the explanation generated for a user who has never heard about them". Lacave and Díez [2002] again recognise that explainability is a graded notion but go further and define the concept of *fixed user model*, noting that practically all explainable BN systems have made this assumption and ignored the possibility of users having varying knowledge. Though, some of the systems surveyed in the paper make a step in that direction by incorporating an *importance threshold* mechanism that would let the user only display certain items; this enables these systems to display varying levels of detail without having defined a user model.

Bayesian Networks, in the taxonomy set forth by Doran et al. [2018] and discussed in Sec. 2.2, would most probably fall into the class of *Interpretable systems*, as do many other ML models. Though, based on the review of the literature, it could be suggested, on quite a strong basis, that Bayesian Networks are better equipped than other Machine Learning models to provide a meaningful explanation to humans. This could be claimed because it is quite widely believed that our brains and thus our psychology are near-optimal problem-solvers and thus approximate optimal Bayesian solutions. A standard view in the fields of psychology and neuroscience, as noted by Bowers and Davis [2012], is that our brain processes approximate the *rational player* as presented in the Dutch Book Argument (see Ch.7 of Anand et al. [2009]) and are thus Bayesian in nature. It is of worth to also note that this view has recently been challenged, for example by Bowers and Davis [2012]. Even if our brains were not inherently Bayesian, the characteristics of Bayesian Networks make them more capable than other ML models in being able to generate explanations tailored to our cognitive processes and psychology, as discussed in the above bullet points.

2.6 Explaining the Most Probable Explanation

The paper "Explaining the Most Probable Explanation" by Butz et al. [2018] inserts itself into the literature concerned with the explainability of Bayesian Networks. In particular, taking Lacave and Díez [2002]'s classification presented in Sec. 2.5, it attempts to define a *verbal explanation*

to the *knowledge base* and of the *propagated evidence*. Unlike Lacave and Díez [2002]'s definition of explanation of a knowledge base and of other previous works, the paper is not concerned with finding the most probable assignment of variables that would explain the given evidence but, rather, the inverse problem. By starting with evidence and finding a maximally probable configuration, the authors hope "to look at the complete scenario to get an overview before deciding which variables should be focused on"; that is, the goal appears to give the user an overview of the situation.

The initial claim of the paper is that even though BNs provide a graphical structure to the knowledge base, they remain of difficult interpretation for domain experts. The examples brought to justify the claim are that edges in the graph do not necessarily represent causal dependencies and that d-separation (defined in 3.4.3) may be confusing. The authors plan to address this claim by constructing a *dialogue* with the user and thus to continue in the long tradition of dialogical approaches to explaining BNs.

The defining characteristic of their approach is that the domain expert is able to "argue" with the MPE and investigate alternative explanations. The complete methodology, that is executed over three steps, is shown in Fig. 2.4. The first step is the construction of the "knowledge base", which is nothing else than a probability tree representing a "chain of deduction" constructed following the strongest probabilistic dependencies between variables in the BN. Such a knowledge base is convenient because the document plan for the Natural Language Generation step is directly derived from it. One issue that is immediately apparent is that this greedy approach does not "generate the MPE solution" as the authors claim, as noted in Subsec. 3.5.2. This does not discredit the argumentative method as a whole, as it is not necessary for the user to be arguing the MPE to derive a good explanation; this ties into one of the main findings in the previous sections that many xAI researchers are only focusing on half of what an explanation is. A good explanation is not given only by its formal properties but, most importantly, by how it acts as an interface between the real *user* and the model. This is what Abdul et al. [2018] mean when they lament that "despite their mathematical rigor, these works *[referring to the existing explainability methods]* suffer from a lack of usability, practical interpretability and efficacy on real users."

The document plan for the argumentation follows the same chain of strongest dependencies constructed in the knowledge base until the expert disagrees; at that point, the user is presented with an alternative "MPE". An example of how the document plan may look after interaction with the user is shown in Fig. 2.5. All the natural language phrasing is generated via boilerplates that realise both the micro-planning phase and the generation of the text.

The authors recognise that such chains of deduction could become long and cognitively overloading in the case of larger BNs, as every variable in the tree is explained by all its ancestors. A solution they propose is that of *pruning* the probability tree by excluding d-separated nodes and those under a certain threshold of significance. They also adapt some methods from literature to perform *conflict analysis*; that is, only variables that contribute positively to the explanation are maintained in the document plan.

On the whole, the paper offers a compelling explanation method for BNs by building on an established tradition of explainability through dialogue. The work, though, takes some methodological missteps and also continues the "sin" of not validating its claims on real users, that is one of the primary gaps in the xAI field that were identified in the previous sections.

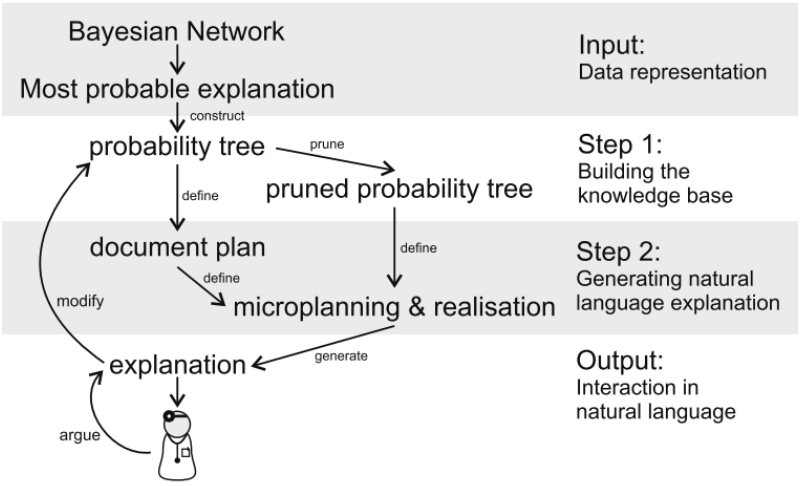


Fig. 2. Overview of the methodology

Figure 2.4. Butz et al. [2018]

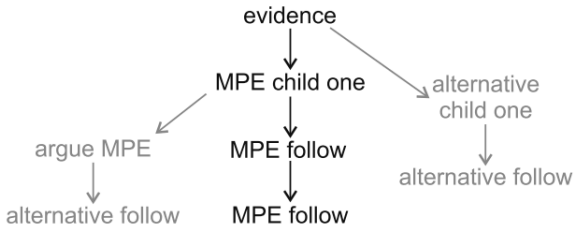


Fig. 4. The document plan derived from the argumentative probability tree

Figure 2.5. Butz et al. [2018]

2.7 Summary

The findings outlined in this chapter refer to the concept of explainability of Machine Learning models, to its importance, to ways of evaluating it and to explainability in the specific case of Bayesian Networks. The concept of explainability is central to the field of Explainable AI (xAI), whose goal is to make Machine Learning systems "interpretable". Explainability can briefly be defined as the property of a system that is able to "explain or to present in understandable terms to a human" its outputs. Two main classes of explainable models have been identified by Mittelstadt et al. [2019]: ante-hoc or transparent and post-hoc interpretable ones; the formers are inherently inspectable in their inner workings while the latters are made understandable by way of extra techniques. These two classes have also been refined by Doshi-Velez and Kim [2017] into a four-tier taxonomy consisting of: opaque, interpretable, comprehensible and explainable systems. An opaque system, also know as a "black-box model", is one whose inner workings are not inspectable from the outside; an interpretable system corresponds to an ante-hoc interpretable one; a comprehensible one emits extra information together with its output; an explainable system explicitly outputs a human-understandable line of reasoning aimed at clarifying its workings.

Explainability has become a central concept to the field of AI as a whole; as ML models take over more and more functions in our societies, the pressure for them to be able to explain their decisions is increased accordingly. The General Data Protection Regulation (GDPR) that became effective in 2018 was viewed by many as increasing the societal pressure to make systems explainable; many may have been mistaken as regards the actual rules mandated by the regulation - that aren't really prescribing a broad "right to an explanation" (Edwards and Veale [2018]) - but nonetheless the feeling of urgency is sure to increase the focus of both researchers and laypeople. In general, explainability is framed as an issue of moral necessity as it easy to find a long series of situations where ML models displayed covert bias or what we would regard as bad moral judgement.

There are all manner of ways to measure explainability and these can be classified into a three-layer taxonomy (Doshi-Velez and Kim [2017]) based on the assumption that the best type of evaluation is the one that most involves humans. The three classes are Functionally-grounded Evaluation, Human-grounded Evaluation and Application-grounded Evaluation, ordered from the one least involving real humans to the one where the presence of the human-in-the-loop is greatest. As the involvement of humans in evaluating models' explanations increases, so does the cost of such an experiment and its specificity, as the highest evaluation level necessarily entails the collaboration of domain-experts on specific tasks. A parallel taxonomy identifies: methods to explain black box models, methods to explain black box outcomes, methods to inspect black boxes and methods to design transparent boxes. An overarching notion that has been stressed is that *explainability is a graded notion* that depends on the knowledge and expertise of the particular user: different users, with different expertises and backgrounds, may rate the quality of a same explanation very differently.

Bayesian Networks (BNs) have enjoyed widespread appeal in mission-critical domains like that of medicine and thus the drive to develop methods to explain their outputs has always been strong. BNs have three main elements that necessitate an explanation (Lacave and Díez [2002]): the knowledge base, the reasoning process and the evidence propagated. Explaining the first "consists of determining which values of the unobserved variables justify the available evidence" and is done by solving the Most Probable Explanation (MPE) problem. For the second, a static explanation of the BN is achieved by displaying it graphically or verbally. The last

element is explained by showing the reasoning that brought the BN to give the outputs it did that can be achieved by providing a justification for its outputs, for the results it did not give or via hypothetical reasoning. The fact that BNs are able to naturally support counterfactual reasoning, combine single variables into composite outputs and model causality puts them at an advantage compared to other ML systems when generating an effective explanation for a user. This is because the capabilities of a BN enable it to generate explanations that are uniquely suited to out psychological biases and expectations of what an explanation should entail.

Some of the main gaps that were found during the review of the literature relate to how there is still a great confusion in the field of xAI regarding what an explanation really is and thus what constitutes a good instance of it. There is also a prevalent methodological confusion, as different authors use terms in incongruous ways, for example sometimes *intepretation* is taken to mean *explanation* while in other cases they refer to different concepts, for example in the taxonomy of interpretable systems proposed by Doshi-Velez and Kim [2017]. This naturally makes it difficult for the field to converge onto methods to evaluate such explanations and this is reflected in the barrage of methods present in the literature, each one focused only on a particular system or instance of model. This confusion is exacerbated by a seeming lack of interest or awareness of xAI researchers for the sizeable corpus of work in psychology, philosophy, social sciences, neuroscience and human computer interaction that has already investigated the nature of explanations, what desiderata they may possess and which are most effective. In fact, the great majority of proposed approaches is only focused on proving formal explainability and neglects the human side that is naturally present in any explanation; there has been little work carried out to validate approaches in real settings with real domain experts so many explainability methods are substantiated only at theoretical level. The underlying issue that I have seen run transversely across the various concepts investigated in this chapter is best summarised by the idea that “inmates are running the asylum”, meaning that individual researchers are claiming that their models are interpretable referring only to their own personal views and biases and not to established literature and methods. It would be hard for them to do otherwise, as the field of xAI seems, at present, to be a collection of diverging strands without a comprehensive program able to help it converge onto its stated goal: to make Machine Learning systems understandable by their users and thus increase their social utility and acceptance.

Chapter 3

Mathematical Background

3.1 Introduction

This chapter will arrive to give a formal definition of Bayesian Networks, a class of Probabilistic Graphical Models that are used to represent systems under conditions of uncertainty. Once the formalism has been defined, an overview of structure learning algorithms and the notions of Conditional Probability and Maximum a Posteriori Query are given. To arrive at this, it will first introduce a series of basic concepts from Probability, Information and Graph Theory. Some of these are not needed for the description of the BN model, but will be useful as a mathematical reference for the work carried out in later chapters of this thesis.

3.2 Probability Theory

3.2.1 Probability Distributions

Definition 3.1 *A probability distribution is a function* $\mathbb{P} : \mathscr{S} \rightarrow \mathbb{R}$ *with* $\mathscr{S}$ *a set of events of interest. To be a valid probability distribution* $\mathbb{P}$ *must satisfy:*

- $\mathbb{P}(\sigma) \geq 0 \quad \forall \sigma \in \mathscr{S}$
- $\sum_{\sigma} = 1 \quad \forall \sigma \in \mathscr{S}$
- $\alpha, \beta \in \mathscr{S} \wedge \alpha \cap \beta = \emptyset \Rightarrow \mathbb{P}(\alpha \cup \beta) = \mathbb{P}(\alpha) + \mathbb{P}(\beta)$

Each event $\sigma \in \mathscr{S}$ must have a probability $\mathbb{P}(\sigma) \in [0, 1]$ and the sum of all these must equal 1. An event with $P(\sigma) = 0$ is deemed *impossible* while one with $\mathbb{P}(\sigma) = 1$ is *certain*.

There is some discord regarding how to actually *interpret* the probability of an event. What I believe to be the initially commonly held view is the *frequentist* one, that views the probability of an event as the ratio of times it would occur over a great number of trials. So, for example, saying that obtaining a heads has probability 0.5 when tossing a coin would mean that over repeated throws we would observe heads half the time.

Another, commonly held view is the *Bayesian* (from the 18th century mathematician Thomas Bayes) one in which probabilities are viewed as the *subjective* degree of belief attributable regarding the manifestation of an event. In this interpretation, stating that a coin has 0.5 probability of landing on heads simply means that the person making the claim personally believes

that the chances of seeing heads of tails are the same. This is obviously a "softer" definition compared to the frequentist one but it is nonetheless useful in that it lets one characterise certain events that haven't come about yet or are liable to happen only once or a few times.

Philosophically, Bayesian inference assigns a probability to a hypothesis (a *prior*) while the frequentist method tests a raw hypothesis empirically before assigning it any probability. As Bayesian inference naturally embraces and deals with uncertainty, it is an enormously useful tool to model and reason about the real, stochastic world we live in.

3.2.2 Random Variables

Definition 3.2 *A random variable is a function that associates every outcome in $\mathscr{S}$ with a value.*

Random variables are a way of bringing to the fore the attributes of interest of events while dealing with them in a clean, mathematical way. The values that a random variable can take are a function of the events in sample space $\mathscr{S}$, each of these is assigned a value by the random variable function. I will only be dealing with *categorical random variables* i.e. those who's codomain is a discrete set of values. Every random variable has a probability distribution induced by the cardinality of the subsets of its values; in the case of categorical-valued one, such a distribution is *multinomial*.

If we were to take a Bayesian point of view, we would consider a random variable as simply representing the subjective degree of belief we would have over a set of outcomes we believed possible.

3.2.3 Conditional Probabilities

After having defined the basic notion of probability, we can construe one the basic building blocks of Bayesian Networks: the concept of *conditional probability*

Definition 3.3 *The conditional probability, "the probability of event β given event α" is:*

$$\mathbb{P}(\beta \mid \alpha) = \frac{\mathbb{P}(\beta \cap \alpha)}{\mathbb{P}(\alpha)} \tag{3.1}$$

That is, the relative proportion of event β compared to event α; this intuitively represents the probability of β *knowing* that α has already occurred.

Equation 3.1 can be easily manipulated to obtain another basic element of Bayesian Networks: what is called the *chain rule of conditional probabilities*:

$$\mathbb{P}(\beta \cap \alpha) = \mathbb{P}(\beta \mid \alpha)\mathbb{P}(\alpha) \tag{3.2}$$

This can be generalised to any number of events:

$$\mathbb{P}(\alpha_1 \cap \ldots \cap \alpha_n) = \mathbb{P}(\alpha_n \mid \alpha_1 \cap \ldots \cap \alpha_{n-1}) \ldots \mathbb{P}(\alpha_1 \mid \alpha_2)\mathbb{P}(\alpha_1) \tag{3.3}$$

Intuitively, it means that we can decompose joint probabilities as products of conditional probabilities. This is how the probability values in a Bayesian Network are calculated.

Another immediate, and crucial, consequence of Eq. 3.1 is known as *Bayes' Theorem*, that lets us calculate the probability of event α given our knowledge of event β.

Definition 3.4 *Bayes' Theorem states that the probability of even α conditional on even β is given by:*

$$\mathbb{P}(\alpha|\beta) = \frac{\mathbb{P}(\beta|\alpha)\mathbb{P}(\alpha)}{\mathbb{P}(\beta)} \tag{3.4}$$

Intuitively, this describes the probability of an event, based on some prior knowledge of another event. For example, our belief in a person having breast cancer (α) is increased if we know that the person is older than 50 years of age (β).

3.2.4 Independence

Now, we have just seen in Equation 3.1 that, in general, $\mathbb{P}(\beta|\alpha) \neq \mathbb{P}(\alpha)$ because $\mathbb{P}(\beta \cap \alpha) \neq \mathbb{P}(\beta)\mathbb{P}(\alpha)$

Definition 3.5 *Two events α and β are unconditionally independent $A \perp B$ - or simply independent - when:*

$$\mathbb{P}(\beta \mid \alpha) = \mathbb{P}(\beta) \Leftrightarrow \beta \perp \alpha \tag{3.5}$$

This means that knowing that α took place doesn't change our beliefs around β happening. In the real world it is hard, or actually impossible if we consider existence at a fine-enough level to involve Chaos Theory, to find two such perfectly non-interacting events. Thus, a more useful concept is that of *conditional independence* where two previously dependent event become independent when also conditioned on a third one.

Definition 3.6 *Two events α and β are conditionally independent $(\beta \perp \alpha \mid \gamma)$ when:*

$$\mathbb{P}(\beta \mid \alpha \cap \gamma) = \mathbb{P}(\beta \mid \gamma) \Leftrightarrow (\beta \perp \alpha \mid \gamma) \tag{3.6}$$

3.2.5 Correlation

Correlation, as defined by Stolp et al. [2006], is a measure of the degree to which two random variables are linearly dependent. The most used measure of such a dependence is the *Pearson Correlation Coefficient* or *bivariate correlation*.

Definition 3.7 *The correlation coefficient ρ of random variables X and Y is given by:*

$$\rho_{XY} = \frac{Cov(X,Y)}{\sigma_X \sigma_Y} \tag{3.7}$$

$$= \frac{\mathbb{E}[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y} \tag{3.8}$$

$$= \frac{\sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} (x-\mu_X)(y-\mu_Y) p_{XY}(x,y)}{\sqrt{\sum_{x\in\mathcal{X}} (x-\mu_X)^2 p_X(x)} \sqrt{\sum_{y\in\mathcal{Y}} (x-\mu_Y)^2 p_Y(y)}} \tag{3.9}$$

with p_{XY} the joint probability mass function of X and Y and p_X and p_Y the marginal distributions of X and Y, respectively.

That is, the bivariate correlation coefficient for random variables X and Y is given by the *covariance* of X and Y divided by the product of their standard deviations.

The covariance is the *first centred moment* of the *joint distribution* of X and Y while the *standard deviation* is the square root of the *second centred moment* of the marginals.

ρ_{XY} is normalised so its values vary in the interval $[-1, 1]$; the correlation coefficient represents the degree of linear association between the two variables with $\rho_{XY} = -1$ being called *perfect anticorrelation* and $\rho_{XY} = +1$ *perfect correlation*. The two correspond to the cases where the linear equation perfectly describes the relationship between X and Y; the sign indicates the slope of the regression line describing the relationship i.e. if an increase in one of the two variables corresponds to an increase in the other in the pair, or viceversa. The closer ρ_{XY} tends to 0, the feebler the relationship between X and Y with the case $\rho_{XY} = 0$ indicating that the two variables are *independent*.

3.3 Information Theory

The birth of the field of *information theory* is usually traced back to the seminal paper "A Mathematical Theory of Communication" (Shannon et al. [1949]) where Claude Shannon set the mathematical basis for the quantification of the amount of *information* transmissible over a noisy channel. In his words "The fundamental problem of communication is that of reproducing at one point, either exactly or approximately, a message selected at another point." The concepts of field are broad enough to have influenced practically every other scientific discipline and deep enough to have enabled the "digital age", for example by enabling the creation of ever more complicated coding schemes for the compression, reconstruction and obfuscation of digital data.

3.3.1 Entropy

In classical mechanical statistics, entropy can be seen as a measure of the uncertainty, or randomness, of a physical system. This concept was reapplied by Shannon to measure the amount of randomness in a random variable.

Definition 3.8 *Given a random variable X with probability distribution $\mathbb{P}(X)$, its entropy $H(X)$ is defined as the expected amount of information content carried by X (Schneider [2005]):*

$$H(X) = \mathbb{E}(I(X)) = \mathbb{E}(-\log(\mathbb{P}(X)) = -\sum_{i=1}^{n} \mathrm{P}(x_i) \log_b \mathrm{P}(x_i) \tag{3.10}$$

The base b of the logarithm defines the unit of measure. Shannon used $b = 2$ as he was dealing with the transmission of digital, binary-coded data; in this case the unit of measure are $bits$.

The simplest example of how information entropy characterises a random variable X, is in imagining X to model a coin and the task being to predict the probability of the outcome of a throw being heads. If the coin is fair, we will not be any more surprised to see the outcome being heads than tails; the entropy is maximum as there is maximum uncertainty regarding the outcome. However, if the coin is not fair and tails is more probable the we will be more surprised than not to see the outcome being heads. The entropy is sub-maximal because there is less uncertainty regarding the outcome: tails is more probable than heads. If one of the outcomes is impossible, for example if the coin has two heads, then the entropy of the coin is 0 as there is no uncertainty regarding the result of a toss.

3.3.2 Normalised Entropy

Plain entropy is not a good choice when trying to characterise random variables with different cardinalities of their sample space. Let us suppose that the objective is to find the variable with the least "entropic" distribution and we suppose that their values have all been generated by the same process, say Gaussian. Simply calculating their entropies and ordering them according to this criterion will bias the selection process towards the variables with smallest cardinality. This is because we supposed them to be distributed in the same way so there will naturally be less uncertainty when there are fewer possible outcomes. This can easily be understood by imagining the distributions to all be random uniform.

To obviate to this problem we need to *normalise* the entropy so that different-sized variables can be directly compared to each other. To achieve this, we can look at a measure of *normalised entropy* or *efficiency*:

$$\eta(X) = -\sum_{i=1}^{n} \frac{p(x_i)\log_b(p(x_i))}{\log_b(n)} \tag{3.11}$$

From Eq. 3.11 it can be seen that $\eta(X) \in [0,1]$; it is thus normalised and comparable among distributions. This ratio expresses the amount of entropy found in the distribution compared to the maximum possible entropy when using n symbols, corresponding to the uniform distribution:

$$\mathrm{H}\left(\underbrace{\frac{1}{n},\ldots,\frac{1}{n}}_{n}\right) = -\sum_{i=1}^{n}\frac{1}{n}\log_b\left(\frac{1}{n}\right) = -n\cdot\frac{1}{n}\log_b\left(\frac{1}{n}\right) = -\log_b\left(\frac{1}{n}\right) = \log_b(n) \tag{3.12}$$

3.3.3 Mutual Information

Another way of characterising the interrelatedness of two variables is through the concept of *mutual information*, as defined in Cover and Thomas [2006], that is closely linked to entropy (Eq. 3.10).

Definition 3.9 *The mutual information of two random variables X and Y is given by:*

$$I(X,Y) = \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p_{XY}(x,y)\log\left(\frac{p_{XY}(x,y)}{p_X(x)p_Y(y)}\right) \tag{3.13}$$

NB: In Information Theory, the convention is that $0\log(0) = 0$.

I_{XY}, intuitively, measures the amount of information that X and Y share that can also be seen as the degree to which one variable is informative of the other. If X and Y are independent then they share no mutual information and knowing one of the two gives no information about the other. This can be immediately understood by rewriting Eq. 3.13 as:

$$I(X,Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) \tag{3.14}$$

The mutual information I(X,Y) is the reduction in uncertainty of one of the variables given the knowledge of the other. if X and Y are perfectly correlated ($\rho_{XY} = \pm 1$) then they both convey the same amount of information and I_{XY} is equal to the entropy $H(X) = H(Y)$.

3.3.4 Hamming Distance

The *Hamming Distance* is a widely-used distance measure that quantifies the similarity of strings.

Definition 3.10 *The Hamming Distance $d_H(x,y)$ between two vectors x and y is given by:*

$$d_H(x,y) = \sum_i \Gamma(x_i, y_i) \tag{3.15}$$

with $\Gamma(i,j)$ defined as:

$$\Gamma(i,j) = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases} \tag{3.16}$$

Given strings of characters, or more in general vectors over some field, of equal length, their Hamming Distance is the number of positions where they differ. It can be seen as the number of substitutions needed to transform one into the other.

This is a valid distance measure because:

- it is non-negative: $d_H(x,y) \geq 0 \quad \forall x,y$
- it fulfils the identity of indiscernibles: $x = y \Rightarrow d_H(x,y) = 0$
- it respects the *Triangle Inequality*: $d_H(x,y) \leq d_h(x,z) + d_H(z,y)$

For example, strings $x = 01234$ and $y = 15244$ have $d_H(x,y) = 2$, as they differ in two positions.

3.3.5 Jaccard Distance

The *Jaccard Distance* is a popular metric to measure the similarity of sets.

Definition 3.11 *The Jaccard Similarity Coefficient $J(A,B)$ of two sets A and B is given by:*

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \tag{3.17}$$

Definition 3.12 *The Jaccard Distance $d_J(A,B)$ between two sets A and B is given by:*

$$d_J(A,B) = 1 - J(A,B) \tag{3.18}$$

This is a valid distance measure because:

- it is non-negative: $d_J(A,B) \geq 0 \quad \forall A,B$
- it fulfils the identity of indiscernibles: $A = B \Rightarrow d_J(A,B) = 0$
- it respects the *Triangle Inequality*: $d_J(A,B) \leq d_J(A,C) + d_J(C,B)$

For example, the sets $A = \{0,2,3,4,1\}$ and $B = \{1,3,5\}$ have $d_J(A,B) = \frac{2}{6}$, as their intersection is of cardinality 2 and their union of cardinality 6.

3.4 Graph Theory

Many problems in Machine Learning (ML) don't involve classification or prediction of single data points in isolation, but of set of entities that may present a more, or less, complex relation with each other. Most real-world phenomena fit into the latter framework. Graphs are one of the most powerful tools for the modelling of this class of problems, as their structure naturally captures the wide variety of relations that may exist between entities. These range from the atomical structure of a molecule to a social network of friends. In both these examples graphs help in reasoning, visualising and making inferences and predictions.

3.4.1 Graphs

Definition 3.13 *A graph is a tuple*

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}) \tag{3.19}$$

with $\mathcal{V} = \{v_1 \dots v_n\}$ *the set of vertices and* $\mathcal{E} = \mathcal{V} \times \mathcal{V}$ *the set of edges.*

For our scopes, we will only be considering the case where every element in $\mathcal{E}$ is a pair either of the form (v_i, v_j) or (v_j, v_i) with $i \neq j$. That is to say that the class of graphs presently of interest for us are those where there can be at most a single directed edge between any node in $\mathcal{V}$ and no self-loops. We are also interested in enforcing that there be no *cycles* in the graph, i.e. sequences of nodes of the form $v_i \to v_j \to \dots \to v_i$. The resulting graph possessing only directed edges and no cycles is commonly called a *directed acyclic graph*, or DAG for short. This data structure is of paramount importance as it's the fundamental graphical representation used for Bayesian Networks.

An example of a DAG, containing five nodes, is shown in Fig. 3.1.

3.4.2 Polytrees

We now have all elements to be able to formally define a Bayesian Network. I will also define polytrees and trees because these are a fundamental concept for the work carried out in this thesis.

Definition 3.14 *A loop is a trace* $v_i, v_j \dots v_i$ *of nodes obtained by following edges regardless of their direction*

Definition 3.15 *A directed graph containing no such loops is called a polytree.*

Definition 3.16 *A tree is a particular case of polytree where each node has at most one parent.*

3.4.3 D-separation

Dependence-separation or *d-separation*, as the name entails, is a concept relating to the conditional dependence between variables. It was first presented by Pearl and Dechter [1988] To define it, we first have to clarify when two sets of nodes X and Y are causally connected. This is so if $Z = \emptyset$ and they are part of one of the following three structures, called *v-structures* in this context:

- $X \to Z \to Y$

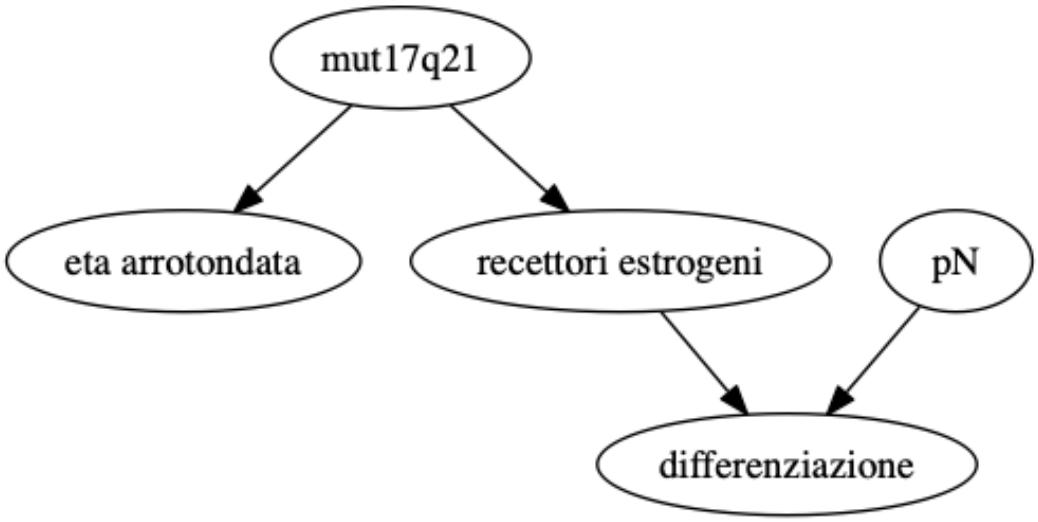


Figure 3.1. Example DAG representing a subset of the data set used in this thesis

- $X \leftarrow Z \leftarrow Y$
- $X \leftarrow Z \rightarrow Y$

This means that knowing something about X also tells us something new about Y. X and Y are causally independent if they appear in the following v-structure:

- $X \rightarrow Z \leftarrow Y$

Such a configuration is called a *collider* and it blocks the flow of information from X to Y. If $Z \neq \emptyset$ the cases are reversed so colliders are open and the other three structures are blocked.

Definition 3.17 *Given disjoint subsets $X, Y, Z \subset \mathcal{X}$, X and Y are d-separated if:*

- *$Z \neq \emptyset$: no path between X and Y presents a collider*
- *$Z = \emptyset$: there is a collider on every path between X and Y*

The independencies between variables are encoded in the structure of the DAG so every distribution whose BN has the same connections between nodes also has the same independencies, regardless of the values of the variables.

A series of examples using the DAG presented in Fig. 3.1 are shown in Fig. 3.2, 3.3, 3.4. We can see how the network's topology and the nodes chosen to be in the observed set Z, define the resulting separations. In all cases $X = \{\text{eta arrotondata}\}$ and $Y = V \setminus X \setminus Z$; we are asking for the set of all nodes in the DAG that are d-separated from X, given evidence Z. The reason for this can easily be done by enumerating all paths through the v-structures

in the network and applying the definitions for causal connections given above. In the case shown in Fig. 3.2 we see that the node **eta arrotondata** is separated from nodes **recettori estrogeni**, **differenziazione** and **pN** given the observed evidence **mut17q21**. The reason for this is because **eta arrotondata** ← **mut17q21** → **recettori estrogeni** is a *fork* and thus the flow of information from the rest of the network is blocked. The way in which changing the conditioning set Z also changes the independencies, can clearly be seen in Fig. 3.3 and 3.4.

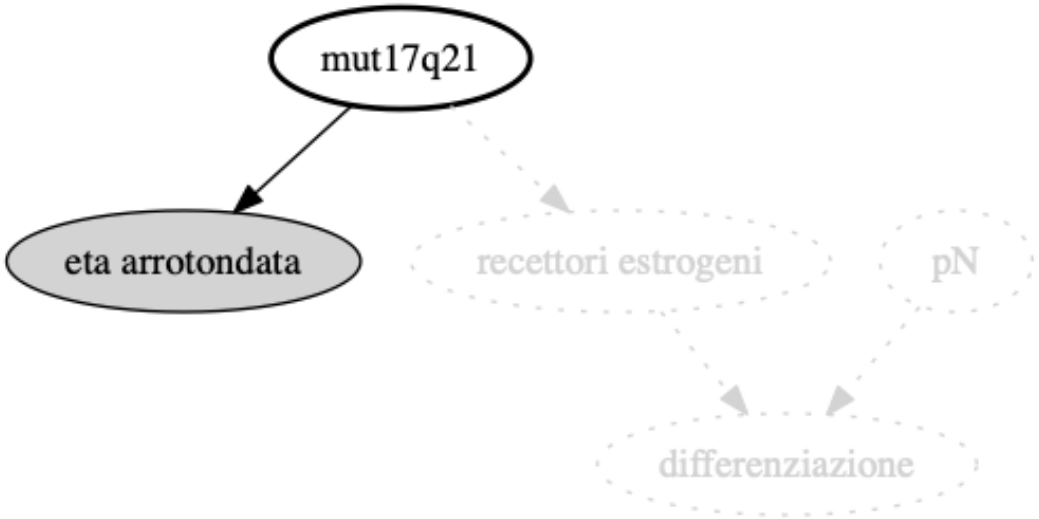


Figure 3.2. D-Separations in a subset of the provided data set (see Sec. 4.2)

3.5 Bayesian Networks

Definition 3.18 *A Bayesian Network (BN) is a probabilistic graphical model represented by a DAG where each vertex corresponds to a random variable X_i and the edges model the dependencies among these.*

Such a model is basically a way of compactly representing an explicit joint distribution $\mathbb{P}(X_1 \cap \ldots \cap X_n) = \mathbb{P}(X_1)\ldots\mathbb{P}(X_n)$, that is factorised into $\mathbb{P}(X_n \mid X_1 \cap \ldots \cap X_{n-1})\ldots\mathbb{P}(X_2 \mid X_1)\mathbb{P}(X_1)$. The way this compactness is achieved is in exploiting the independencies that exist among the random variables:

$$\forall X_i : (X_i \perp \neg Desc(X_i) \mid Pa(X_i)) \tag{3.20}$$

with $Pa(X_i)$ the set of nodes that are parents of X_i and $Desc(X_i)$ the nodes that are not *descendents* of X_i. That is to say, every random variable X_i, given its parent nodes, is independent of all other nodes in the Bayesian Network that are not descended from it. Also, a BN gives the

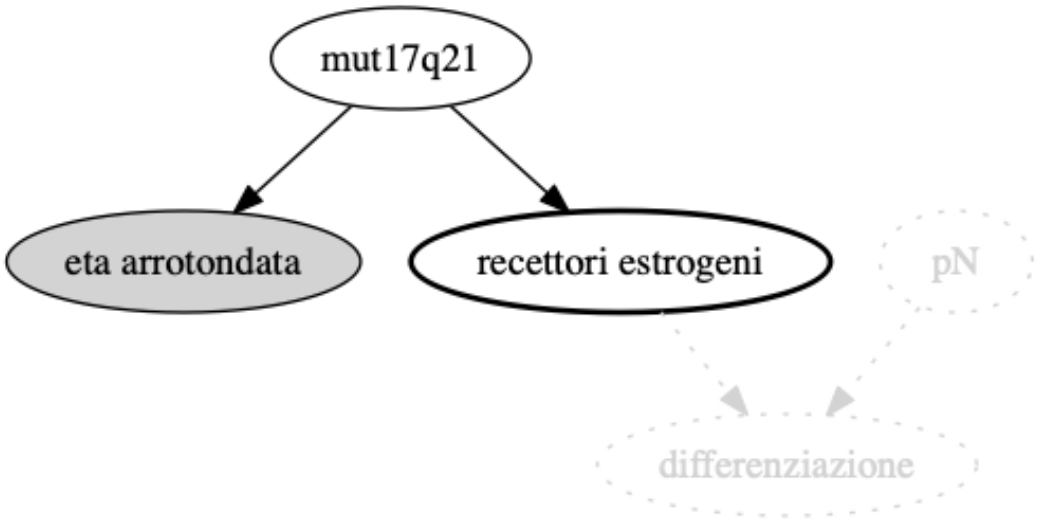


Figure 3.3. D-Separations in a subset of the provided data set (see Sec. 4.2)

flexibility to drop the many weak dependencies that are bound to exist between variables thus leading to an even simpler model. A full probability table for a joint distribution of random variables obscures the independencies and requires an exponential number of entries for the representation. A Bayesian Network on the other hand can represent the same distribution using only a linear number of parameters. The way that Bayesian Networks can be used to reduce the storage requirements for uncertain information is by taking advantage of the conditional independencies embedded in the underlying distribution being modelled. The power of BNs comes from the additional information encoded in their structure and this was first explicitly described in its entirety by Pearl and Dechter [1988] who defined the concept of dependence separation (see Subsec. 3.4.3) and applied it to Bayesian Networks.

One nice characteristic of BNs is that they very naturally model the type of mixed causal and stochastic processes that we find in all of Nature. Imagine we want to represent the process modelled by joint distribution $\mathbb{P}(B,A) = \mathbb{P}(B)\mathbb{P}(A)$; using the chain rule for conditional probabilities (Eq. 3.3) we can write this as $\mathbb{P}(B \mid A)\mathbb{P}(A)$. A BN modelling this process would be composed of two nodes A and B with an edge from the former to the latter $A \rightarrow B$, A is called the "parent" of B. Each of these two nodes would have its own probability table, with $\mathbb{P}(A)$ representing the *prior* distribution over A and $\mathbb{P}(B \mid A)$ the *conditional probability distribution* of B given A.

We can now see why these types of models are named *Bayesian* Networks: the inference process is based in a given prior distribution/belief and evolves through a parent $\rightarrow$ child relationship to constantly yield an updated *posterior* belief. The BN DAG encodes a generative sampling where each variable's value is determined stochastically by Nature, based on the value

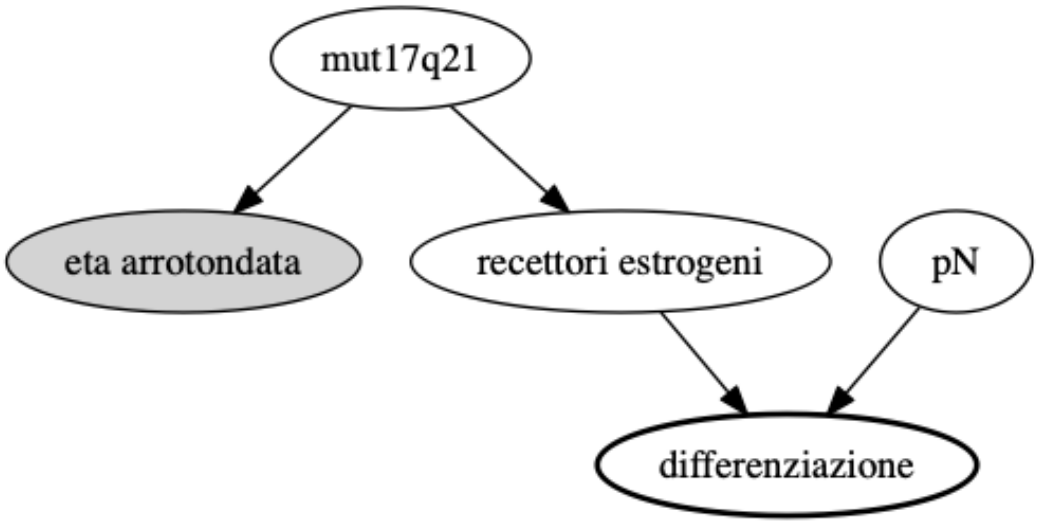


Figure 3.4. D-Separations in a subset of the provided data set (see Sec. 4.2)

of its parents. This process is also highly compatible with our view of causality and this is one of the reason that makes BNs highly interpretable. The prior $\mathbb{P}(A)$ can be seen as the result of some stochastic process caused by a series of latent (unmodelled) variables while the posterior $\mathbb{P}(B \mid A)$ is stochastically, causally determined by A. As I have mentioned in the previous paragraphs, there are probably no truly "prior" distributions in the Universe, at the modelling scale we are usually interested in. Only on arriving on the quantum particle level may we find "pure" stochastic, uncaused processes due to quantum collapse.

A good example of how BNs are well compatible with our notion of causality may be to imagine A as the random variable modelling the predisposition to having a certain disease and B to actually developing the symptoms for it. *First*, genetic and epigenetic factors such as the environment stochastically contributed to having the predisposition and *then* the development of the symptoms was stochastically determined by the degree of predisposition. Adding an extra time dimension certainly helps us in dealing with this class of models.

The example show in Fig. 3.1 is the underlying graph structure of a Bayesian Network, each node is now representing a Random Variable with an associated *Conditional Probability Table* (CPD), that defines its probability distribution, conditional on its parents. The CPDs for **eta arrotondata** and **mut17q21** in the Bayesian Network in question are shown in Tab. 3.1 and 3.2. **Mut17q21** is a root node, i.e. has no parents, in the DAG so its probability distribution is unconditional or *marginal*. **Eta arrotondata**, on the other hand, is a child of **mut17q21** so the probability of its values is conditional on that of its parent. For example, **eta arrotondata** takes on value "<40" 44% of the time when **mut17q21** has value "mut", but only 4% of the time when **mut17q21** has value "unknown".

Table 3.1. mut17q21 CPT

	mut	unknown
mut17q21	0.006	0.99

Table 3.2. eta arrotondata CPD

		mut17q21	
		mut	unknown
eta arr.	<40	0.42	0.04
	40-50	0.42	0.17
	>50	0.15	0.78

3.5.1 Bayesian Networks Structure Learning

In many probabilistic models initialisation is fast but then fitting the data is slow (ex. k-means). For Bayesian Networks the converse is true: fitting is fast as only sums of the counts in the data are needed but identifying the correct graph structure can take super-exponential time. Learning the Bayesian Network structure from data is commonly known as the Bayesian Network Structure Learning (BNSL) problem. The methods to solve this problem can be roughly categorised into one of three types.

Search and Score

This is the most naïve method as it does a brute force search over all the possibile graph structure space - i.e. all DAGs with the same number of variables as the input data - and scores all these depending on some cost function. This process is super-exponential but though the use of dynamic programming and heuristic search algorithms it can become sub-exponential. Nonetheless, solving the exact BNSL is only feasible up to 30 variables.

Constraint Learning

Methods of this type calculate some measure of correlation to identify the presence and direction of edges between nodes. A typical test is to iterate over all triplets while testing for conditional independencies. Thanks to the d-separation properties outlined in Subsec. 3.5, this test is able to identify the correct edges. The algorithm is quadratic in time in the number of vertices.

Approximations

Several heuristical approaches have been developed to be able to find good network structures in an efficient manner. Examples of these are:

- Chow-Liu, that builds a tree approximation of the probability distribution
- Greedy Hill-Climbing, that adds/removes/flips an edge at a time

- Optimal Reinsertion, that iteratively calculates the optimal $Markovblanket$ (the subset of all nodes that are sufficient to determine the value of another subset) of an ever-smaller subset of nodes

3.5.2 Bayesian Networks Updating

All the types of inference presented are instances of *diagnostic reasoning*, also known as *abductive reasoning*. This type of explanation can either be modelled as a conditional probability or a MAP query and is of fundamental importance in many important problems of machine learning including medical diagnosis, that is of particular interest to us.

Conditional Probability Query

The *updating* problem is the process of updating the probabilities of nodes in the BN based on the observation of the values of other vertices. This process of conditioning on observed information is also called *data propagation*.

The following algorithm was described by Normand and Tritchler [1992] and applies to our case where the random variables follow a multinomial distribution. What we want, is to calculate the conditioned probability $\mathbb{P}(B \mid D)$ i.e. the updated probability of node B based on observed evidence E.

Definition 3.19 *The conditional probability query for variable B given evidence E is:*

$$\mathbb{P}(B \mid E) = \alpha\pi(B)\lambda(B) \tag{3.21}$$

with $\pi(B)\lambda(B)$ analogous to the prior and likelihood of B, respectively.

The likelihood of B depends only on the weighted likelihoods of its children $C_1, \ldots, C_k$:

$$\lambda(B) = \prod_l \lambda_{C_l}(B) \tag{3.22}$$

$$\lambda_{C_l}(B) = \sum_{C_l} \lambda(C_l) P(C_l \mid B) \tag{3.23}$$

and its prior similarly depends only on the information received from its parents A:

$$\pi(B) = \sum_A P(B \mid A)\pi_B(A) \tag{3.24}$$

$$\pi_B(A) = \alpha\pi(A) \prod_{S_B} \lambda_{S_B}(A) \tag{3.25}$$

The information is propagated down if any variable observed is above B while up if any variable observed lives in the tree rooted in B. Initially all leaf nodes' likelihoods are set at 1 and the priors of root nodes are assumed to be observable.

Maximum a Posteriori Query

Another common type of question we might ask a BN is the following: "given evidence E which is the most likely assignment of a subset of variables Y?". This is know as *Maximum a posteriori (MAP)* inference and is a much harder problem that a conditional probability query. We are trying to solve the an optimisation problem.

Definition 3.20 *As defined by Koller et al. [2009a]. Given evidence/observed variables $E = e$, $E \subseteq \mathcal{X}$ and sets $Y \subseteq \mathcal{X} - E$ and $Z = \mathcal{X} - E - Y$, with $\mathcal{X}$ the set of all variables in the BN, the MAP query for Y is the assignment of values $Y = y$ that has maximum probability:*

$$MAP(Y = y \mid E = e) = \underset{y}{argmax} \sum_{z} \mathbb{P}(Y = y, Z = z \mid E = e) \tag{3.26}$$

The MAP problem is hard to solve efficiently; that is it is part of the *NP-hard* complexity class, as proved by Shimony [1994]. Calculating it in a brute-force way would mean elencating all the possible variable-value tuples and computing their joint probabilities; as these are exponential in the number of variables, the problem is evidently untractable. Moreover, this is true even in a Bayesian Network. Such a model may possess a linear number of parameters but the underlying distribution is still exponential. Explicitly calculating the MAP defeats the very purpose of the BN, that is computational efficiency. For this reason, there exist a host of approaches to optimising MAP: elimination algorithms, gradient methods, simulated annealing and other stochastic local searches, belief propagation and integer linear programming.

A very important thing to note is that the greedy assignment where each variable picks its most likely value can be very different from the most likely joint assignment of all variables.

Most Probable Explanation Query

A special case of MAP is the *Most probable explanation (MPE)* that,

Definition 3.21 *As defined by Koller et al. [2009a]. Given evidence/observed variables $E = e$, $E \subseteq \mathcal{X}$ and $W = \mathcal{X} - E$, the MPE query for W is the assignment of values $W = w$ that has maximum probability:*

$$MPE(W = w \mid E = e) = \underset{w}{argmax}\, \mathbb{P}(W = w \mid E = e) \tag{3.27}$$

This is an easier problem than MAP, as can be seen by comparing Eq. 3.26 with Eq. 3.27; MAP presents both a summation and a maximisation and as such is part conditional probability query, part MPE query. All algorithms for the computation of MAP obviously apply to MPE too, but there exist efficient approximate algorithms for MPE that do not generalise to MAP such as Loopy Belief Propagation (Pearl and Dechter [1988]) and Stochastic Local Search (Kask and Dechter [1999]).

3.6 Summary

The chapter has introduced a number of concepts from Probability, Information and Graph Theory to be used as groundwork for the formal description of Bayesian Networks that are a widely used class of probabilistic graphical models.

The section dealing with Probability Theory opened by describing the fundamental concept of *probability distribution* that is a function from a set of events of interest to the real numbers. The probability of an event may be interpreted through the *frequentist* or the *Bayesian* lens; the former sees probability as simply the limit of the ratio between the number of times the event of interest occurred and the total number of trials; the latter views probabilities as the subjective degree of belief of the manifestation of the event. A *random variable* is a mathematical construct

that associates a value to every outcome in the set of possible events and is used to bring to the fore the attributes of interest while dealing with them in a clean way. The main results presented are *Bayes' Theorem*, that states how to update a prior belief in light of new knowledge, and the concept of *independence* between events, that will be central when introducing *d-separation*. The final concept presented in the Probability Theory section is that of Correlation, a universally accepted measure for the interrelatedness between random variables.

The second section introduces a few key concepts from Information Theory relating to entropy and distance measures. The *Entropy* is a measure for the expected amount of information carried by a random variable, first introduced by Claude Shannon under inspiration by mechanical statistics. A derived quantity is *normalised entropy*, also known as *efficiency*, that is convenient because it varies between 0 and 1 and thus enables random variables and probability distributions of different sizes to be compared. A second method, closely related to entropy, of measuring the interrelatedness of two random variables is that of *mutual information*; this measure quantifies the amount of information of one variable already contained in the other. Two popular distance measures were introduced: *Hamming and Jaccard Distance*; the former measures the similarity of strings, based on the number of substitutions needed to transform one into the other, and the latter quantifies the similarity of sets, given the size of their intersection over union.

The third section relates to Graph Theory and starts by defining the basic notion of *graph*, a set of vertices and edges connecting them, and of *Directed Acyclic Graph*, a special case of graph whose edges are directed and that contains no cycles between vertices. *Trees* and *polytrees* are briefly introduced and characterised as a particular case of DAG. Finally, *d-separation*, first introduced by Judea Pearl, is discussed. D-separation is a concept relating to the conditional dependence between variables; sets of variables may become independent i.e., not influence each other, based on conditioning on a third set of evidence variables. The independence properties depend on the topology of the graph, specifically in how the variables of interest are connected to each other; they may be organised into *chains*, *forks* or *colliders*.

The final section of the chapter deals with introducing *Bayesian Networks*, using many of the concepts laid out in the previous sections. A Bayesian Network is a probabilistic graphical model represented by a DAG where each vertex corresponds to a random variable and the edges model the dependencies among these. The basic idea is to factorise a complete joint distribution of the constituent variables into a series of Conditional Probability Tables, one for each variable, that are assigned to the nodes in the DAG. The defining characteristic is that each variable's node values depend only on those of its parents. Such a representation efficiently represents a joint distribution and very naturally models the type of mixed causal and stochastic processes found in Nature. The DAG of a BN can either be given or learned directly from data; in the latter case, that is a super-exponential problem, there are three main classes of algorithms that may be applied: Search and Score, Constraint Learning and Approximations. Once a DAG has been learned, the problem moves to querying (updating) the BN; the main classes of queries are Conditional Probability and Maximum a Posteriori Queries. The first class asks for the value of a set of variables given the observation of the values of others in the network. The second class, known as MAP queries, ask the question of finding the most probable assignment of values to a subset of variables, given the observation of the values of another subset. This is, in general, an NP-hard problem but efficient solutions exist to a special case known as the Most Probable Explanation, where the set of query variables in the complementary subset to the evidence one.

Chapter 4

Methodology

4.1 Introduction

The inspiration for the work carried out in this thesis was the paper "Explaining the Most Probable Explanation" by Butz et al. [2018], that has been reviewed in detail in Sec. 2.6. The paper proposed a system that, starting from a Bayesian Network modelling a medical data set, would learn a "knowledge base" tree representing the chain of most probable deductions, starting from a set of initial evidence. This tree, deemed to represent the solution to the MPE query, could then be used to generate a dialogue in natural language with the medical expert, that the authors claim could lead to the extraction of extra knowledge from the original data set. The driving hypothesis of the paper was that Bayesian Networks and the solution to the MPE problem would be a powerful tool in helping medical experts gain insights into data.

The paper did not provide any indication that a such a system had ever been built and any validation of the method was left by the authors for future work. This lack of real-world validation has been seen, as discussed in Chap. 2, to, unfortunately, be the norm in most papers published under the Explainable AI moniker. Many works are content to only give a *Functionally-grounded Evaluation* (in the taxonomy of Doshi-Velez and Kim [2017]) for the methods they propose. Butz et al. [2018]'s does not even give such an evaluation of the methods it proposes. As of the finalisation of this thesis (August 4, 2019), there has been no work carried out in substantiating Butz et al. [2018]'s conclusions. As introduced in Chap. 1 and discussed in detail in Sec. 2.3, there is an ever greater need for Machine Learning models and systems to be explainable, especially in mission-critical domains as healthcare.

For these reason the belief is that building a proof-of-concept system, whose logic is inspired by the method presented in the aforementioned paper, and validating it with real medical experts, will prove to be an important step forwards in the direction of providing the work with the highest level of corroboration, an *Application-grounded Evaluation*. The objective of this thesis is not only to provide an assessment of the paper, but also to set a methodological precedent for the evaluation of a Machine Learning system with real domain experts on real tasks. This, as has been discussed in Chap. 2, is one of the main gaps existing today in the field of xAI; thus, carrying out such an evaluation presents a substantial element of novelty. Finally, it is hoped that the proof-of-concept system may be of real use to the medical experts who were provided with it, in performing their crucial work.

The chapter opens with an introduction of the partner involved in the evaluation of the meth-

ods: *Istituto Cantonale di Patologia* (ICP), Locarno, that specialises in the histological analysis of tissue samples. The data set, that was provided as base for the methods of this thesis, is comprised of the clinical profiles of 3218 breast cancer patients in the Swiss canton of Ticino.

The chapter continues with a presentation of the technological tools used to build the proof-of-concept system that was given to the ICP. The main ones of interest are *Pomegranate*, an open-source probabilistic models package for Python, Pgmpy, another graphical model package for Python and DAOOPT, a specialised solver for the MPE problem that takes as input the `.uai` format, also described in the section. Two graphical model packages were used because there does not yet exist a Python library that implements in a convenient way all the functionality needed to work with Bayesian Networks.

Next,

4.2 Data set

As anticipated in Chap. 1 the work carried out in this thesis had a certain degree of collaboration with a third party, Istituto Cantonale di Patologia.

4.2.1 Istituto Cantonale di Patologia

Istituto Cantonale di Patologia (ICP) is an institute based in Locarno that is specialised in the histological analysis of tissue samples received from private patients, clinics and hospitals, mainly in support of cancer diagnosis. Its Laboratory of Medical Diagnostics supports pathologists in the diagnosis of neoplastic diseases through the application of cytogenetic techniques; that is, the focus is on understanding how chromosomes relate to cell behaviour, particularly during mitosis and meiosis. One of the techniques used is Fluorescence in Situ Hybridization (FISH) that is able to localise the presence or absence of specific DNA sequences in chromosomes. These tests are aimed at identifying the precise profile of the cancer cells and thus inform the clinician on the best treatment for the specific patient.

In addition to its clinical support activities, the Istituto also carries out scientific research aimed at better understanding certain types of cancers at a basic level. In the last ten years, the ICP has published more than 200 peer-reviewed papers and more than 100 works in non-peer reviewed journals and is active at a national and international level.

4.2.2 Motivation

The Istituto Cantonale di Patologia already had a collaboration in place with the Dalle Molle Institute for Artificial Intelligence (IDSIA) (ids) to investigate a series of specific issues, whose details are outside of the scope of this thesis. The work carried out in this thesis was deemed of interest because its scope went beyond the existing collaboration. The institute had thus expressed interest in bringing machine learning into their workflow in order to both augment its profiling capabilities for patients and to be able to extract new knowledge from their existing data, together with more experimental directions as the facilitation of human-machine interaction.

magari togliere CV e mettere descrizione

chiedere a Vittoria cosa si aspettavano/aspettano

My interest in collaborating with the Istituto stemmed from the desire to apply the methods described in Sec. 4.4 to a real-world case. My first contact with the ICP was during a meeting

with Dr. Vittoria Martin (Martin [2012]), molecular citogenetist, in date 28/01/2019. Being the theoretical work carried out in this thesis an expert-driven MPE approximation, collaboration with the institute also provided the opportunity to implement a proof of concept system using a real histological data set. The clinicians and researchers of the Istituto have been able to validate, from an Explainable AI and clinical relevance point of view, the model software that I have developed. That is to say, they have validated, to an extent that will be made clear in Chap. 5, the capabilities of the developed software both in its capacity to support clinical decision making and to surface clarifying explanations of the data set and in the adherence of its outputs to established medical literature.

An example application for a clinician of the ICT, would be ability to "fill in the blanks" of a patient's profile, as it is not uncommon, for a variety of reasons, to have missing data. This may be because of degraded or insufficient tissue samples or because some test may not yet be part of the standard diagnostic procedure, even though their importance may already be suggested by clinical research. In other cases, patients may be missing a result because the specific test hadn't yet been invented, for example FISH wasnâĂŹt available prior to 2010, so an a-posteriori inference could be made possible. These are all examples highlighting the importance of the *inference* capabilities of a Machine Learning system, but my current work aims to address the interfacing of the human user with the software. It is also hoped that facilitating the process of knowledge-extraction may lead towards the confirmation of current scientific theories or may be the first step towards the formulation of novel ones.

aspettare idee Ginevra su ulteriori applicaz cliniche

da confermare con Vittoria piu avanti

4.2.3 Provided Data Set

The data set I was provided with was created by *Registro Tumori Ticino* (Locarno, Ticino) in order to highlight possible new relations between clinical, histopathological and molecular features, as well as to potentially discover novel biomarkers for the progression of the disease It consists of the histological records over 38 recorded variables of 3218 breast cancer patients who have been diagnosed between the years 2005 and 2014 within the Ticino canton of Switzerland. The data set had been pre-processed by collaborators of IDSIA in agreement with the ICT with 13 of the variables being dropped, because not considered relevant. In particular, all variables relating to the patient post-treatment were discarded as well as those recording the diagnosis date. In Tab. 4.1 is a description of the measured variables, together with their clinical meaning. The value distribution of the data set is shown in Tab. 4.2.

The indications from Dr. Martin on how to further preprocess the data are shown in Tab. 4.3. Note that some variable names were simplified.

4.3 Methods

4.3.1 Libraries

Pomegranate

Pomegranate (Schreiber [a]) is an open-source probabilistic models package for Python. Its core philosophy is that every probabilistic model, from Hidden Markov to Bayesian Network, can be seen as a probability distribution and, as such, can be flexibly composed into hierarchical mixture models (Schreiber [2017]). The package implements:

Table 4.1. Data set variables

Variable	Clinical meaning
Codice globale	Unique patient identifier
mut17q21	Mutation of chromosome 17
loss 17	Loss of chromosome 17
età arrotondata	The age of the patient at diagnosis
Lateralità	The affected breast
Situ SUBGROUP MZ	The primary site code of the tumour
Morfologia SUBGROUP MZ	The morphology classification of the tumour
pT SUBGROUP MZ	Primary tumour in the TNM classification for breast cancer
pN SUBGROUP MZ	Pathologic in the TNM classification for breast cancer
M 8.2.96	Distant metastasis in the TNM classification for breast cancer
Differenziazione	Tumour grade
Recettori estrogeni percento 1.1.2003	Expression of estrogen receptors
Recettori progestinici percento 1.1.2003	Expression of progestin receptors
c erbB 2 cod percento 1.1.2003	ErbB2 marker expression
Ki67 cod percento	Tumoural proliferation index
FISHRatio	FISH analysis result

- Probability Distributions
- General Mixture Models
- Hidden Markov Models
- Bayes Classifiers and Naïve Bayes
- Markov Chains
- **Bayesian Networks**
- Factor Graphs

This package was chosen among others for its good implementation of Bayesian Networks and its performance. The package is written in cython and natively supports multi-core parallelism and out-of-core learning. Network structure learning from data, described in 3.5.1, appears to be particularly efficient, thanks to the implementation of prior knowledge into the graph selection process as described by Schreiber and Noble [2017]. The claim of this novel selection process is that it possesses the speed of a heuristic approach while yielding a far better quality estimate.

Pomegranate currently only supports Discrete Bayesian Networks so the random variable of each node must have a categorical distribution.

Structure learning from data is achieved using the `from_samples` method of the `BayesianNetwork` class, with the default algorithm being the novel one described by Schreiber and Noble [2017].

Table 4.2. Data set distribution before pre-processing

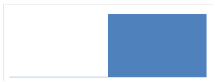
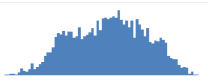
Variable	Unique values	Distribution
mut17q21	2	
loss 17	3	
eta arrotondata	74	
ateralità	3	
situ	5	
morfologia	5	
pT	23	
pN	6	
M	3	
differenziazione	5	
recettori estrogeni	40	
recettori progestinici	40	
c erbB 2	4	
Ki67	52	
FISH	5	

Table 4.3. Data set preprocessing steps

Variable	Action
Codice globale	Remove variable
mut17q21	Remove blanks
loss 17	Remove blanks
eta arrotondata	Bin into “< 40”, “$40 - 50$”, “≥ 50”
lateralita	Remove blanks and “sconosciuta”
situ	Remove blanks
morfologia	Remove blanks and “unuseful” if performance on classification is subpar
pT	Remove blanks and “unuseful”
pN	Remove blanks and bin into “0” and “$\neq 0$”
M	Remove blanks
differenziazione	Remove blanks and “Sconosciuto o non applicabile”
recettori estrogeni	Remove blanks and bin into “negativo” if ≤ 10, “debolmente positivo” if ≤ 50, “fortemente positivo” if > 50
recettori progestinici	Remove blanks and bin into “negativo” if ≤ 10, “debolmente positivo” if ≤ 50, “fortemente positivo” if > 50
c erbB 2	Remove blanks
ki67	Remove blanks and bin into “<14”, “14-20”, “20-30”, “>30”
FISH	Remove blanks

The *probability* of a sample is calculated using the `probability` function of an object of `BayesianNetwork` type; the `predict_proba` function is used to return the probability of each variable in the model given some evidence. *Predictions* (described in detail in Subsec. 3.5.2) are run by passing to the `predict` function of an object a matrix with `None` as placeholders for missing values . *Fitting* is done thought the `fit` function that uses MLE estimates to update each node's distribution in the model based on the input data.

A `BayesianNetwork` object can also be displayed graphically by calling its `plot` function. The output is a DOT file that is generated using the PyGraphviz package (PyGraphviz developer team), that is a python interface to the famous Graphviz (gra) graph visualisation software. An example of such an output is shown in Fig. 4.1.

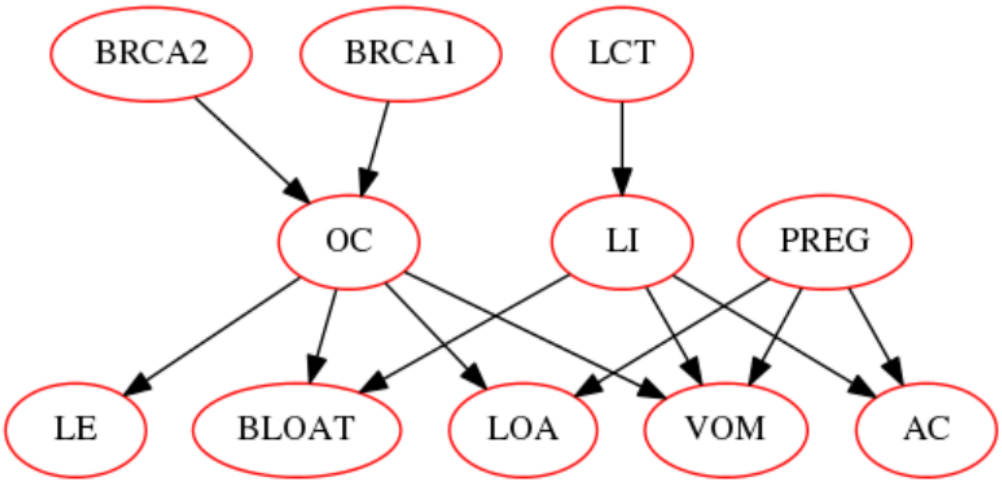


Figure 4.1. Example output of `plot` (Schreiber [b])

Pgmpy

Pgmpy (pgm) is, like pomegranate, another recent probabilistic graphical model package for Python. Unlike pomegranate, it natively implements various exact and approximate inference algorithms, like Variable Elimination, Belief Propagation and Max-Product Linear Programming.

The reason that I elected to use two different probabilistic graphical model libraries is because there is still no Python package that offers all the needed functionality. Pomegranate implements a novel structure learning algorithm, that I wanted to use, but is severely lacking in functionality in many other areas. Pgmpy, on the other hand, has a very good API as regards inference.

DAOOPT

DAOOPT is an open-source implementation of sequential AND/OR Branch-and-Bound proposed by Marinescu and Dechter [2006]. Search-based algorithms traverse the model's search space and are much more efficient in their use of space, compared to inference-based algorithms such as Variable Elimination.

DAOOPT builds an AND/OR search space to generate search an AND/OR graph that takes advantage of information encoded in the graphical model, namely its independencies. The DAOOPT implementation found at dao is an exact solver for finding an MPE solution in Bayesian Networks. The solver is written in C++ and accessible through a command-line interface; the only required parameter is a `.uai` file representing a Markov Random Field or a Bayesian Network but in most cases an optional `.uai.evid` file will also be given, containing the observed evidences.

UAI file format

The `.uai` file format is a simple text file used to represent problem instances. Such a file is composed of:

- Preamble: containing the type of the network (MARKOV or BAYES), the number and cardinality of variables and the cliques, that in the BAYES case are simply the variables appearing in each Conditional Probability Table (CPT)
- Function tables: containing the actual definition of the CPTs i.e. the values of each node give its parents or, in the case of root nodes, the marginal probabilities.

The `.uai.evid` is a very simple file containing the number of variables in the evidence set followed by the index of each variable and its observed value. In both formats the variables and their values are represented only by a numerical index, starting from 0, with the ordering being defined in the preamble of the `.uai` and maintained consistent throughout the `.uai` and `.uai.evid`.

Following, is the `.uai` representing the network shown in Fig. 3.1, that has been our running example throughout the last chapters. Lines starting with `c` are interpreted as comments; these are misinterpreted by DAOOPT and are thus removed when running it, but are here shown for clarity. The file starts by stating that the model is a Bayesian Network composed of 5 random variables that will then be referenced by an ordinal index starting at 0; the first variable (index 0) is of cardinality 3, the second (index 1) is of cardinality 2 and so on. We can then see the definition of the cliques or more precisely, as our model is a BN and not a MRF, of the CPTs; there are 5 of these, each one associated to one of the five variables just stated. The first CPT involves 2 random variables: the first (0) and the second (1); the second CPT involves only one variable (1) and this tells us that variable 1 is a root node in the BN's DAG. The ordering is such that the child node is the last in the definition of each CPT's nodes so, for example, in the first CPT we find that variable 1 is the child of variable 0. Finally, we have a complete definition of the function tables/CPTs. The tables are printed so that each row corresponds to the conditional probability value of the child node and increasing rows correspond to increasing enumeration of the parents' states, in the order given when defining the variables involved in the CPTs. The first table corresponds to **eta arrotondata**'s CPT shown in Tab. 3.2. It contains 6 elements as it involves variables 0 (**mut17q21**) and 1 (**eta arrotondata)** that are of cardinality 2 and 3, respectively. So each row corresponds to the probability distribution of the three states of variable 1, given each of the two states of variable 0.

```
c
c Bayesian Network exported from pomegranate - Thomas Tiotto (2019)
c
```

```
BAYES
5
3 2 3 3 2

c
c Cliques
c

5
2 0 1
1 1
2 2 1
3 3 2 4
1 4

c
c CPTs
c

6
 0.42105263157894735 0.42105263157894735 0.15789473684210523
 0.043798177995795384 0.17063770147161877 0.7855641205325858

2
 0.006613296206056387 0.9933867037939436

6
 0.6842105263157895 0.0 0.3157894736842105
 0.1373510861948143 0.021723896285914507 0.8409250175192712

18
 0.004385964912280701 0.2412280701754386 0.7543859649122807
 0.022598870056497175 0.11864406779661016 0.8587570621468926
 0.10344827586206899 0.41379310344827586 0.4827586206896552
 0.2121212121212121 0.45454545454545453 0.3333333333333333
 0.14094488188976378 0.6362204724409449 0.22283464566929131
 0.289612676056338 0.5677816901408451 0.1426056338028169

2
 0.5315001740341107 0.4684998259658893
```

The following is an example of a randomly generated `.uai.evid` evidence file that simply states that the evidence set has cardinality 2 and contains variable 4 (in the ordering given in the `.uai`) in its state 1 and variable 3 in state 2.

```
 2
  4 1
  3 2
```

These were generated by a function I have written (see 4.3.2) that is able to export a `pomegranate` model and randomly generated evidence to the correct imput format for DAOOPT.

Gurobi

Gurobi (gur) is a closed-source mathematical programming solver for Linear Programming, Quadratic Programming and Mixed-Integer Programming optimisation problems. It claims to be the fastest solver available for these classes of problems. Gurobi offers object-oriented and matrix-oriented interfaces to, among others, Python, MATLAB and Excel.

Pandas

pandas (pan) is an extremely widely-used open-source python library that provides data structures and methods to aid in data analysis. The package excels in the manipulation of tabular data in the form of `DataFrame`, that is the analogous of R's `data.frame`. A `DataFrame` can be seen as a "general 2D, size-mutable structure with potentially heterogeneously-typed columns". The syntax for slicing is very close to R's as are many other functionalities; this is because one of Pandas' explicit goals is to offer all of CRAN's functionalities and be easily approachable by anyone already knowing the other language.

Pandas the default choice for this thesis' implementation because it is the *de facto* standard in data analysis applications. Its flexibility in reading Excel spreadsheets (the format the data set the project was built on, see Sec. 4.2) and in then manipulating the data confirmed that this was a good choice. Note that to read files in the Excel formats the additional `xlrd` package is needed.

Scikit-learn

scikit-learn (sci) aims at providing a unified API for basic Machine Learning; it does not include advanced paradigms such as Reinforcement Learning or graphical models for structured learning. The latter omission was the reason that lead me to select pomegranate as the basis for the implementation of a Bayesian Network. What is included are a stack supervised and unsupervised ML tools to prepare data sets, define machine learning models ranging from spectral analysis-based to ensemble methods to clustering and multiple evaluation and model selection utilities.

NumPy

NumPy (num) is another *de facto* standard package when doing scientific computing with python. Most scientific packages (including pandas, scikit-learn and TensorFlow) depend on NumPy for low-level operations; this is because NumPy is contains fast implementation of n-dimensional array objects together with powerful manipulation functions. In addition to this, NumPy implements linear algebra operations, Fourier Transform and random number generation. The closest parallel to NumPy - as R was for pandas, is MATLAB.

Networkx

NetworkX (net) is another widely-used package that is specialised in the creation and manipulation of graph-structured data. The main use for this package was in building the `MPEGraph`

data structure that I used to build a polytree representing the dialogue with the expert.

4.3.2 Algorithms

quanti dettagli implementativi devo metter chÃ¨ ho un sacco di funzioni di visualizzaz etc

Model construction

The data was given in `.xlsx` format and was imported using `panda`'s `read_excel` function that returned a `DataFrame` object. The imported data was then preprocessed by dropping unwanted records and binning the remaining ones as described in Tab. 4.3 . The actual BN representation is learned at runtime by calling the `from_samples` method of `pomegranate`'s `BayesianNetwork` to solve the structure learning problem (defined in Subsec. 3.5.1).

The binned data was codified into integer representations before being passed to pomegranate's structure learning algorithm. Thus the network's state names are in natural language but the internal representation of the values of each random variable is an integer number. A dictionary object is used to translate one representation into the other at runtime.

d-separation

A naïve implementation according to the definition (presented in Subsec. 3.4.3) to check for d-separation between node X and Y would have a complexity in the order of the number of trails between X and Y; this would lead to an exponential in the size of the graph running time. Luckily, Koller et al. [2009b] present a linear time algorithm to solve the problem.

The `reachable` procedure takes as input the DAG representing the Bayesian Network $\mathcal{G}$, a source variable X and a set of observed variables Z; on exit it returns the set of variables R that are reachable from X. The procedure runs in two phases, traversing the graph twice: first bottom-up from leaves to roots, then viceversa. During the first run, the algorithm finds all nodes A that are ancestors of the evidence set Z. During the second, the procedure distinguishes the direction it visits each node in order to determine if it is traversable or not. Any node Y that is not in the evidence set is marked as reachable; if it is being visited in direction "up" it can be traversed as the v-structure is a chain. All the parents of Y are marked to be visited in the "up" direction (i.e. from below) and the converse is done for Y's children. If Y is being visited in the "down" direction its children are again added to be visited in the "down" direction, because Y is traversable. Additionally, if Y happened to be in the set A, found in the first step, then Y's parents are marked to be visited in the "up" direction because the collider is active and Y can be traversed (a collider is open iff. the central node or any of its descendants are observed).

The full procedure can be found in Koller et al. [2009b]; my implementation follows this pseudocode very closely but the procedure `d-separated`, instead of finding all nodes R that are d-connected to the input X, tests if a given target Y is d-separated from X or not, as is shown in Alg. 1. This gives some extra flexibility in how the function can be used. To find the set S of all nodes d-separated from X I simply iterate the `d-separated` test over all nodes V in the graph representing the BN.

MPE

The solution to the Most Probable Explanation problem (MPE), as defined in Subsec. 3.5.2, is found by using DAOOPT (Subsec. 4.3.1) as an external solver. The latest version of DAOOPT

Algorithm 1 d-separation algorithm

1: $separated\_list = \emptyset$
2: **for** target $Y \in V$ **do**
3: append $d-separated(X, Y, E)$ to $separated\_list$ ▷ will return true or false
4: **end for**

was downloaded from the github page (dao) and compiled into a executable. DAOOPT only offers a command line interface so some extra work was needed in order to interface it with the Python-based application I have developed. The connection was done by first writing to stable storage a `.uai` containing the model definition and a `.uai.evid` with the observed evidence. These files are then piped to DAOOPT by using Python's `subprocess` module to run the following command in an external background shell:

```
./daoopt -f pomegranate.uai -e pomegranate.uai.evid
```

The shell output is captured and also written to stable storage, in order to be decoded and used in the system.

To exemplify the process, we return to the example used while presenting DAOOPT in Subsec. 4.3.1. Given the `.uai` representing the BN and the `.uai.evid` random evidence:

```
2
 4 1
 3 2
```

DAOOPT would give the following output:

```
--- Starting search ---
[0] u 3 4 -1.3581 5 2 1 2 2 1
[0] Cache statistics: . . . .

--------- Search done ---------
Problem name:  pomegranate
OR nodes:      3
AND nodes:     4
OR processed:  3
AND processed: 8
Leaf nodes:    2
Pruned nodes:  4
Deadend nodes: 1
Time elapsed:  0 seconds
Preprocessing: 0 seconds
-------------------------------
-1.3581 (0.0438433)

p 2 1 2
l 2 1 6
s -1.3581 5 2 1 2 2 1
```

The end of the final line is the one of interest as it is the assignment of values to the variables that solves the MPE problem. The `5 2 1 2 2 1` string is to be interpreted as meaning:

- there are 5 variables in the solution
- the variable indexed by 0 (in the ordering given in the preable of the `.uai`) is assigned its second value (the ordering is inferred by the CPTs defined in the `.uai`) in the MPE solution
- variable 1 is assigned its second value
- variable 2 is assigned its third value
- variable 3 is assigned its third value
- variable 4 is assigned its second value

Variables 3 and 4 are constrained to assume the value specified in the input `.uai.evid`; in this case 1 and 2, respectively.

All the functionality relating to solving the MPE with DAOOPT is encapsulated in the `daoopt_solver` function that given the input `.uai` files, returns the MPE solution.

Other Machine Learning methods

In order to quantify the prediction capabilities of the Bayesian Network, that is at the heart of this thesis' methods, I implemented a series of tests to find the best performing algorithm on the data set. Given that the performance of Machine Learning algorithms is heavily dependent on the input classes, I used a process of *exhaustive variable elimination* in order to identify the most relevant features for the predictions. In parallel, I scored each input subset using the following ML algorithms, in order to find the best performing one:

- Linear Regression: this method assumes that the relationship between the dependent variable y and the regressors x is linear i.e. that y can be written as a linear combination of x's components: $y = \beta_0 + \beta_1 x_1 + \dots \beta_n x_n$.
- Logistic Regression: is used in lieu of Linear Regression when the values of the variables are categorical; it assumes that the relationship between the regressors x and the log-odds of y are linear i.e. $\log_b \frac{p}{1-p} = \beta_0 + \beta_1 x_1 + \dots + \beta_n x_n$ with p the probability of the event of interest.
- Linear Discriminant Analysis: LDA is related to Principal Component Analysis (PCA) in that it attempts to find a linear equation modelling the data but LDA explicitly tries to express the difference between the data classes.
- Decision Tree: a Decision Tree is built using a recursive, greedy algorithm that continually splits the dataset into two. The variable along which to bisect is the one that yields the lowest accuracy loss in the resulting split.
- Naïve Bayes: a Naïve Bayes classifier is a conditional probability model that given features $x_1 \dots x_n$, attempts to assign a probability to each of the possible outcomes O_k of interest by using Bayes' Theorem (Eq. 3.4): $\mathbb{P}(O_k|x) = \frac{\mathbb{P}(O_k)\mathbb{P}(x|O_k)}{\mathbb{P}(x)}$. The method is called "naïve" because of a strong (ofter unrealistic) assumption of all the features $x_1 \dots x_n$ being independent.

- K-Nearest Neighbours: the algorithm is non-parametric with the output class depending on the predominant class among the k nearest neighbours (according to some distance metric) of the input vector x.
- Support-Vector: A Support-Vector Machine (SVM) is an algorithm that attempts to find the set of *best-separating hyperplane* between classes of objects, seen as points in a high-dimensional space. Such a hyperplane is the one that has maximum distance from the closest representatives of each class.
- Random Forest: this is an ensemble method that aims to correct Decision Trees' tendency to overfit the data. A multitude of Decision Trees is constructed and the final classification output is the class that appears most often in the intermediate step.
- AdaBoost: Short for *Adaptive Boosting*; this meta-learning, ensemble algorithm combines a series of weak classifiers, that may only be slight better than a random guess, through a weighted sum into a *strong classifier*. It is a meta-learning algorithm because the weak classifiers are revised over a series of iterations in order to improve their performance on previously misclassified instances.

4.4 Novel contributions

tengo qui o sposto nel cap 4?

la discussione con l'esperto ci da' belief rev
dei dati

counterfactual explanations

magari togliere theory e mettere tutto assi

utilizzo entropia

4.4.1 Entropy-Based Selection

esempio fittizio che esemplifica il fatto che usare la probabilitÃă Ãĺbiased verso variabili a cardinalitÃă piÃź bassa cardinalitÃă diversa, grandezze non confrontabili quanto ci si discosta dalla uniforme? information gain, l'uniforme ha sempre entropia 1, con cardinalita piÃź grande a stessa probabilitÃă abbiamo piÃź informazione guadagnata

4.4.2 Algorithms

An important part of my work was developing the algorithms needed to adapt the ideas presented in the paper "Explaining the Most Probable Explanation" by Butz et al. [2018] and "A Progressive Explanation of Inference in 'Hybrid' Bayesian" by **?**. From the former, the construction of the probability tree through a constructive dialogue with the domain expert, the building of counterfactual explanation branches, the automatic generation of the most probable probability tree from initial evidence. From the latter, the generation of an "Inverse explanation". Finally, a simple procedure to output a natural language explanation was developed.

Dialogues

The so-called "Pseudo-MPE" algorithm is inherently wrapped up with the concept of *dialogue* and is central to the explanatory powers of the system being developed in this thesis. The algorithm was developed as a way of implementing the "MPE branch" of the "Argumentative Probability Tree" hypothesised by Butz et al. [2018]. It was termed "Pseudo-MPE" because there

are no guarantees of it returning the MPE solution (see Subsec. 3.5.2), as noted by Koller et al. [2009a] in their definition of the MAP problem.

At a lower level of detail, the algorithm may be broken into:

- a dialogical part, that interfaces with the expert user through the use of natural language, menus and visualisations
- the part responsible for constructing the "MPE branch"

The former process was informed and shaped by the results obtained by the methods described in Subsec. 4.5.2 and, as such, presented substantial elements of novelty.

The latter process is, at its core, a greedy procedure that aims at selecting the "best" next $(state, value)$ tuple at each step, based on some measure of optimality and on the variables already in the evidence set. In the actual implemented system the two parts are intertwined, given their close inter-dependence.

The `dialogue` procedure starts by asking the user to select a subset of variables and their relative values to add as initial evidence. This initial evidence is used to radicate the MPE Branch. It should be noted than in the description given by Butz et al. [2018], the Argumentative Probability Tree is a real tree as each node is guaranteed to have at most one parent. My application, on the other hand, constructs an *Argumentative Probability Polytree* (see Subsec. 3.4.2) because, as will better be described in Chap. 5, it was seen early on that the users much preferred to be able to start from a set of initial evidences and not be limited to a single one. The algorithm then proceeds to call the `next_most_probable_states` subroutine that is tasked with returning an ordered list of $(state, value)$ pairs. It does this by calculating the posterior distribution given evidence of all the states not already in the evidence, then calculating the efficiency (see 3.3.2) and the maximally probable symbol of each state's distribution and finally returning the $(state, value)$ tuples ordered according to their normalised entropy (most efficient/least entropic at the head). The $(state, value)$ pair at the head of the list is proposed to the user who has the faculty to accept the system's evaluation or refuse it. If the user accepts, the state is added to the evidence set and to the MPE Branch under construction. Thus, the evidence set's cardinality increases by one each time a user accepts a proposal. The updated evidence will be used to calculate the new list of $(state, value)$ pairs at the following round. If the expert chooses to refuse, then she is iteratively presented with the remaining $(state, value)$ items, in order of decreasing efficiency. Once she accepts one of the explanations given by the system, the `generate_alternative_branch` subroutine is called to automatically generate a maximally probable MPE Branch, radicated in the newest $(state, value)$ node of the MPE Branch. The proposal loop for alternative states runs until there are increasingly less probable elements in the list and exits with a partial solution if the user refuses all of them at a given step. Thus, the Pseudo-MPE solution is constructed only if the user runs through all variables proposed, accepting each one at least once.

Three slightly different operational modes of the algorithm were implemented. This was done for research purposes, in order to understand which of the three, if any or if a combination of their distinctive features, the expert users would find the most useful from a usability, comprensibility and explainability standpoint:

- exhaustive: In the basic dialogue type, the set of variables under consideration monotonically decreases by one every time the user accepts one of the system's proposals and the dialogue terminates only when the user has accepted all variables at least once or refused all proposals at a given step. In the first case the user will have the Pseudo-MPE solution

while in the second she will be left with a partial assignment to some of the variables not present in the initial evidence. The pseudocode is shown in Alg. 2.

- d-separated: In the second variant, the set of variables under considered at each step is dynamic and depends on the separation properties of the underlying Bayesian Network's DAG and the evidence set constructed by the user's choices. Differently from the first type of dialogue, an additional `evidence_d_separation` subroutine is called before `next_most_probable_states` to calculate the set of variables that are d-separated from the evidence set, up to that step of the dialogue. `next_most_probable_states` is then executed but the variables that the previous function found to be separated from the evidence, are removed from the returned list. This way, variables that can have no effect given the current evidence are not proposed. As the d-separation operation is not monotonic, adding new nodes to the evidence set can both increment or diminish the number of nodes that will be proposed at each step. The user is shown an updated view of the independencies of the graph at each step; an example of such an output is shown in Fig. 4.2 and 4.3. The pseudocode is shown in Alg. 3.

- thresholded: The final variant of the algorithm prunes the set of variables using a different strategy from the previously presented one. In this case, the $(state, value)$ pairs in the list returned by `next_most_probable_states` are dropped automatically based on their probability. Pairs whose probability is below a user-defined threshold or are "worse than random" (for ex. a $(state, value)$ tuple will be discarded if $state$ is binary and the probability of $value$ is lower than 0.5) are removed and not proposed to the user. This thresholding strategy based on the probability of the tuples is paired with one where there is a user-defined threshold on the number of times that the expert can refuse a particular $(state, value)$. In the general dialogue, tuples can be proposed multiple times, with an ever lower probability, if the user has previously refused them; in the thresholded scheme a $(state, value)$ pair can only be proposed a maximum number of times before being permanently discarded.

The underlying Bayesian Network that represents the data set is learned and queried through the `pomegranate` (see Subsec. 4.3.1), API but the great majority of all the code is completely custom-written. This was necessary because `pomegranate`, while having a powerful backend, was found to be severely lacking in the breadth and flexibility of its API. Many basic operations, such as the calculation of a joint distribution, were not available so the only way was to implement lower-level workarounds while still using `pomegranate` for the most basic operations, as the calculation of a posterior distribution. In particular, `dialogue` is implemented with the only direct calls to the API being when learning the network and when calling `predict_proba`, that queries the `BayesianNetwork` object to calculate the posterior distribution of the states given the current evidence. D-separation, in the second variant of the algorithm, is calculated via the `evidence_d_separation` procedure that implements the pseudocode presented in Alg. 1.

Alternative Explanation Branches

The function to generate alternative branches to the main MPE branch in the dialogue tree is called after the user refuses a $(state, value)$ in the dialogue and accepts one of the the alternatives. The motivation is to present the user with a simple "what-if" analysis, replying to the question "Had I accepted the $(state, value)$ presented me by the system, what would have

Algorithm 2 Exhaustive pseudo-MPE algorithm

```
1: evidence = user selected (state, value) tuples
2: MPE_polytree = MPE Polytree rooted in evidence
3: while True do
4:     mpe_states = next_most_probable_states(evidence)
5:     if mpe_states is not empty then
6:         next_state = head of mpe_states
7:         propose next_state to user                          ▷ the least entropic state
8:         if the user refuses next_state then
9:             for alternative_state in mpe_states ∖ next_state do
10:                propose alternative_state to user            ▷ the next least entropic states
11:                if the user accepts alternative_state then
12:                    call generate_alternative_branch() on MPE_polytree
13:                    add alternative_state to MPE_polytree
14:                    evidence = evidence ∪ alternative_state
15:                else
16:                    continue
17:                end if
18:            end for
19:        else
20:            add next_state to MPE_polytree
21:            evidence = evidence ∪ next_state
22:        end if
23:    else
24:        return
25:    end if
26: end while
```

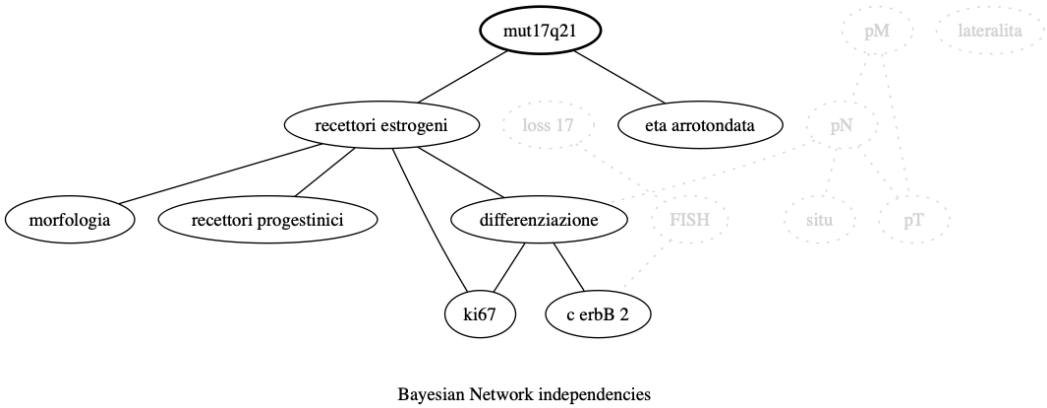


Figure 4.2. Example output during the first round of the d-separation-aware variant of `dialogue`. The variable "mut17q21" is the initial evidence.

Algorithm 3 Independencies pseudo-MPE algorithm

```
1: evidence = user selected (state, value) tuples
2: MPE_polytree = MPE Polytree rooted in evidence
3: while True do
4:     separated = evidence_d_separation(evidence)      ▷ based on evidence of previous step
5:     mpe_states = next_most_probable_states(evidence)
6:     mpe_states = mpe_states ∖ separated
7:     if mpe_states is not empty then
8:         next_state = head of mpe_states
9:         propose next_state to user                    ▷ the least entropic state
10:        if the user refuses next_state then
11:            for alternative_state in mpe_states ∖ next_state do
12:                propose alternative_state to user     ▷ the next least entropic states
13:                if the user accepts alternative_state then
14:                    call generate_alternative_branch() on MPE_polytree
15:                    add alternative_state to MPE_polytree
16:                    evidence = evidence ∪ alternative_state
17:                else
18:                    continue                          ▷ go to next proposal
19:                end if
20:            end for
21:        else
22:            add next_state to MPE_polytree
23:            evidence = evidence ∪ next_state
24:        end if
25:    else
26:        return
27:    end if
28: end while
```

Algorithm 4 Thresholded pseudo-MPE algorithm

```
1: user selected refuse_bound
2: refuse_thresholds = ∅
3: lower_thresholds = ∅
4: for v ∈ V do
5:     for value k of v do
6:         element [v, k] = 0 in refuse_thresholds
7:     end for
8:     element [v] = 1/|v| in lower_thresholds                ▷ refuse worse than random pairs
9: end for
10: evidence = user selected (state, value) tuples
11: MPE_polytree = MPE Polytree rooted in evidence
12: mpe_states = next_most_probable_states(evidence)
13: while True do
14:     for s ∈ mpe_states do
15:         if probability of s < lower_thresholds[s] ∧ refuse_thresholds[s] > refuse_bound
    then
16:             remove s from mpe_states
17:         end if
18:     end for
19:     if mpe_states is not empty then
20:         next_state = head of mpe_states
21:         propose next_state to user                        ▷ the least entropic state
22:         if the user refuses next_state then
23:             increment refuse_thresholds[next_state]
24:             for alternative_state in mpe_states ∖ next_state do
25:                 propose alternative_state to user         ▷ the next least entropic states
26:                 if the user accepts alternative_state then
27:                     call generate_alternative_branch() on MPE_polytree
28:                     add alternative_state to MPE_polytree
29:                     evidence = evidence ∪ alternative_state
30:                 else
31:                     increment refuse_thresholds[alternative_state]
32:                     continue
33:                 end if
34:             end for
35:         else
36:             add next_state to MPE_polytree
37:             evidence = evidence ∪ next_state
38:         end if
39:     else
40:         return
41:     end if
42: end while
```

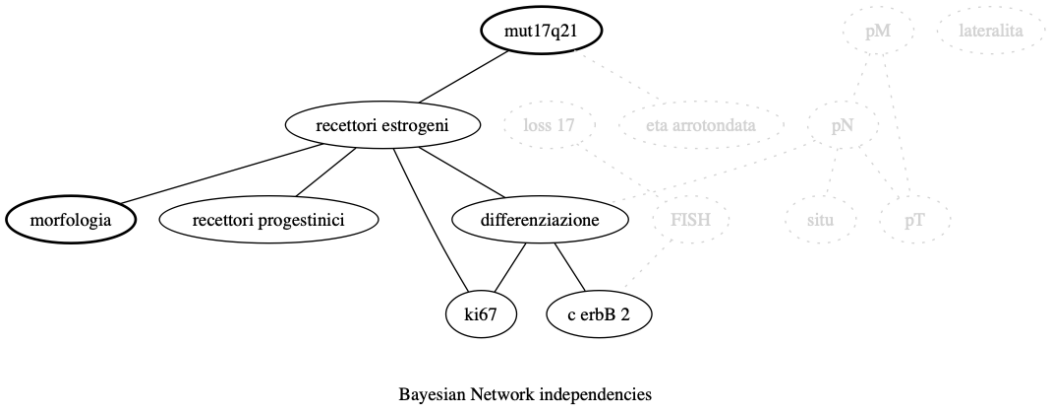


Figure 4.3. Example output during the second round of the d-separation-aware variant of `dialogue`. "morfologia" is added to the evidence set and this makes a large part of the network redundant.

been the most probable configuration of the remaining $(state, value)$ pairs?". The question is answered by generating a maximally probable, alternative MPE sub-branch rooted in the last node in the main MPE branch that is under construction through the dialogue.

The alternative branch is generated by what is essentially an automated version of `dialogue` that always accepts the first suggestion returned by `next_most_probable_states`. Given that `dialogue` and `generate_alternative_branch` are essentially one and the same, the latter inherits the same pruning strategies as the former. That is, `generate_alternative_branch` called during the exhaustive dialogue will generate a maximally likely assignment over all variables in $V \setminus E$ while when invoked from one of the other two variant of the dialogue algorithm, it will apply their same pruning strategies.

The implementation of the MPE Polytree is based on the `NetworkX` python package (see Subsec. 4.3.1). The creation of a chain of nodes is done by keeping a local pointer alt_node that refers to the last added node or set of nodes, if the node being added is the successor of multiple initial evidences.

An example of the output shown to the user at each step of the dialogue is shown in Fig. 4.4 while the pseudocode for the three variants are shown in Alg. 5, 6 and 7. Note that $alternative\_evidence$ is local to this algorithm and is separate from the $evidence$ used in the main `dialogue` procedure.

"Pseudo-MPE" from Initial Evidence

In order to compare the Pseudo-MPE output with the true MPE solution I implemented a simple algorithm that, starting from an initial set of evidence, generates the relative pseudo-MPE solution. The random initial evidence set is constructed by randomly choosing a number in the interval $k = [1, |V|]$, with V the set of vertices in the BN, and then randomly selecting k of the random variables in V to yield the set of variables E. The value of each variable is randomly chosen among the set of its values; as all variables are categorical, this can easily be done.

An optional threshold can be set in order to discard $(state, value)$ pairs whose probability is deemed too low.

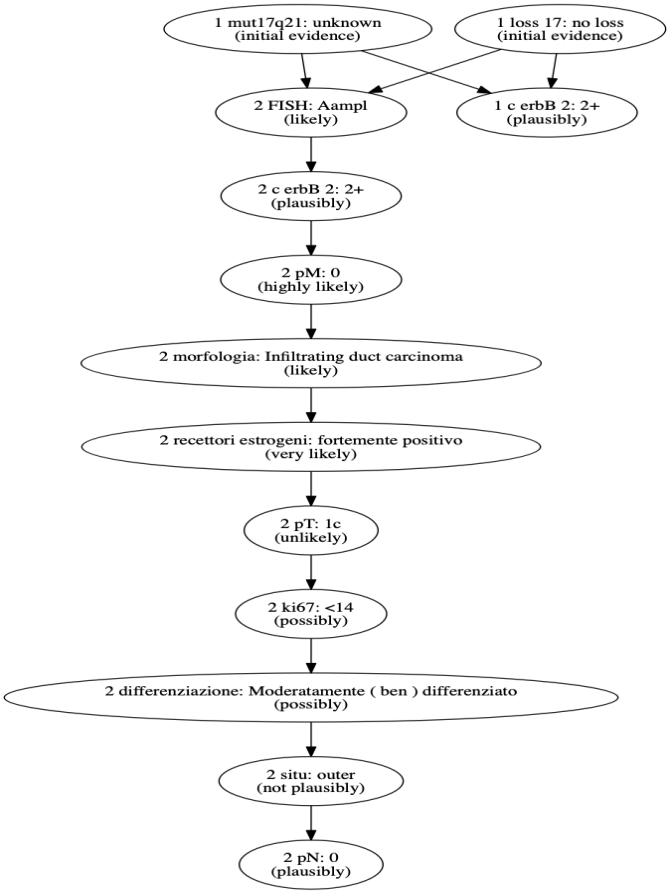


Figure 4.4. Example output during the d-separation-aware variant of `dialogue`. The tuple ("FISH","Aampl") was proposed but the expert refused it and accepted the alternative ("c erbB 2","2+"). The main MPE Branch has ID 1 while the "what-if" one has ID 2.

Algorithm 5 Exhaustive alternative explanation branch algorithm

```
1: alternative_evidence = evidence
2: alt_node = last node in the main MPE Polytree
3: branch_id = branch_id + 1
4: while True do
5:     mpe_states = next_most_probable_states(alternative_evidence)
6:     if mpe_states is empty then
7:         return
8:     else
9:         next_state = head of mpe_states
10:        create next_state node, tag it with branch_id and make it son of alt_node
11:        update alt_node node to be next_state node
12:        alternative_evidence = alternative_evidence ∪ next_state
13:    end if
14: end while
```

Algorithm 6 Independencies alternative explanation branch algorithm

```
1: alternative_evidence = evidence
2: alt_node = last node in the main MPE Polytree
3: branch_id = branch_id + 1
4: while True do
5:     separated = evidence_d_separation(alternative_evidence)
6:     mpe_states = next_most_probable_states(alternative_evidence)
7:     mpe_states = mpe_states ∖ separated
8:     if mpe_states is empty then
9:         return
10:    else
11:        next_state = head of mpe_states
12:        create next_state node, tag it with branch_id and make it son of alt_node
13:        update alt_node node to be next_state node
14:        alternative_evidence = alternative_evidence ∪ next_state
15:    end if
16: end while
```

Algorithm 7 Thresholded alternative explanation branch algorithm

```
1: alternative_evidence = evidence
2: alt_node = last node in the main MPE Polytree
3: branch_id = branch_id + 1
4: while True do
5:     for s ∈ mpe_states do
6:         if probability of s < lower_thresholds[s] ∧ refuse_thresholds[s] > refuse_bound
   then
7:             remove s from mpe_states
8:         end if
9:     end for
10:    mpe_states = next_most_probable_states(alternative_evidence)
11:    if mpe_states is empty then
12:        return
13:    else
14:        next_state = head of mpe_states
15:        create next_state node, tag it with branch_id and make it son of alt_node
16:        update alt_node node to be next_state node
17:        alternative_evidence = alternative_evidence ∪ next_state
18:    end if
19: end while
```

The implementation is based on the `NetworkX` python library (see Subsec. 4.3.1) package as what is being constructed is not a tree but a *polytree* (see Subsec. 3.4.2), as nodes may have multiple parents. Note that $alternative\_evidence$ is considered separate from the main $evidence$ used in `dialogue`.

The pseudocode is shown in Alg. **??** while an example output can be seen in Fig. **??**.

Algorithm 8 Pseudo-MPE from inital evidence algorithm

```
1: evidence defined by user
2: threshold defined by user
3: MPE_polytree = MPE Polytree rooted in evidence
4: last_node = evidence
5: alternative_evidence = evidence
6: while True do
7:     mpe_states = next_most_probable_states(alternative_evidence)
8:     if mpe_states is empty then
9:         return
10:    else
11:        next_state = head of mpe_states
12:        if probability of next_state ≤ threshold then
13:            return constructed MPE_polytree
14:        end if
15:        create next_state node and make it son of last_node
16:        update alt_node node to be next_state node
17:        alternative_evidence = alternative_evidence ∪ next_state
18:    end if
19: end while
```

MPE Algorithms Comparison

In order to compare the quality of the solution found using the simple Pseudo-MPE heuristic, I set up an experiment to compare it to the exact MPE solutions calculated with DAOOPT (see Subsec. 4.3.1). The user is able to select the number of iterations over which to average the results and at each iteration a Pseudo-MPE solution is generated using Alg. **??**. The same initial evidence is fed to DAOOPT using the `daoopt_solver` interfacing procedure shown in Subsec. 4.3.2 and to pgmpy's `map_query` function.

At each iteration corresponds a random evidence; the solutions given by the Pseudo-MPE, pgmpy and by DAOOPT are scored against each other using Hamming (Subsec. 3.3.4) and Jaccard (Subsec. 3.3.5) distances on the vectors representing the returned assignments. The sequence of distances is then averaged over the iterations to return the two averaged distances between the three solutions to the MPE problem.

The implementation of the algorithm had to deal with the fact that DAOOPT failed on certain configurations of input evidence. The reason for this is unknown so my workaround was to discard any iterations where DAOOPT had failed and retry them with another random set of evidences. The pseudocode for the algorithm is shown in Alg. 9.

Algorithm 9 MPE comparison algorithm

user defined number of $iterations$
initialise empty $results$ list
while $i < iterations$ **do**
    generate random number $k \in [1, |V|]$
    $S =$ choose k variables from V
    for $s \in S$ **do**
        choose random v in the possible values of e
        $evidence = evidence \cup (s, v)$
    end for
    $pseudo\_mpe =$ `generate_pseudo_mpe(`$evidence$`)`
    $model\_uai =$`export_model_to_uai()`
    $evidence\_uai =$`export_evidence_to_uai()`
    $daoopt\_mpe =$ `daoopt_solver(`$model\_uai$`,` $evidence\_uai$`)`
    if $daoopt\_mpe ==$ -1 **then** ▷ DAOOPT has failed to calculate a solution
        cancel $pseudo\_mpe$
        continue
    end if
    compute $d_H(pseudo\_mpe, daoopt\_mpe)$, $d_J(pseudo\_mpe, daoopt\_mpe)$ and store in $iter\_result$
    append $iter\_result$ to $results$
    $i+ = 1$
end while
compute average in $results$ for d_H and d_J

Inverse Explanation

da fare e trovare nome migliore

Pairwise Correlations

An interesting addition, in terms of both explainability and theory, is an algorithm to measure and graphically display the interrelatedness between pairs of variables. This is achieved by calculating the *conditional mutual information* between each pair of parent → child variables. The definition of mutual information presented in Subsec. 3.3.3 is extended to account for the current state of the model i.e. the set of observed variables. What we are calculating is:

$$I(X,Y|E=e) = \sum_{x\in\mathscr{X}}\sum_{y\in\mathscr{Y}} p_{XY}(x,y|E=e)\log\left(\frac{p_{XY}(x,y|E=e)}{p_X(x|E=e)p_Y(y|E=e)}\right) \tag{4.1}$$

This is not done if the parent or child variable, in the $X \rightarrow Y$ tuple under consideration, is in the evidence set; this is because observing a variable in a BN conceptually corresponds to disconnecting it from its children.

The implementation takes advantage of pgmpy's (pgm) inference capabilities. To do this, I wrote a function to convert a pomegranate-based BN to an equivalent pgmpy-based one. The queries to the model are done using the variable Elimination algorithm, that is more than suitable for a small BN. The marginals for X and Y are calculated directly from the joint distributions, by marginalising the joint over one and then the other. The function exits with -1 if the parent variable is in the evidence set. As the mutual information $I(X,Y) \in [0,1]$ this signals to the calling function to treat this couple X, Y differently.

The pseudocode for the algorithm is shown in Alg. 10.

4.4.3 Interfacing with the User

Natural language was believed to be the most spontaneous way to make the expert user aware of the knowledge in the data set and for her to interact with the system. Thus the main interaction mode with the implemented system is in using natural language, both as input and as output. Natural language denominators come "for free" when dealing with the node names and their values, regardless of their internal representation in the implemented model, as these are simply the column names and values found in the original input data set.

A bit more care needs to be taken when quantifying probabilities to the user. The format chosen should fit in as seamlessly as possible with the rest of the phrases generated and be able to conform to the user's preconceptions regarding probabilities. It should be both *informative* and *precise*.

The chosen coding was taken from Butz et al. [2018] and is shown in Tab. 4.4. The mapping is used every time there is a need to translate a probability into natural language to be output to the user; for example when proposing tuples during a Dialogue (see Subsec. 4.4.2) as example of which can be seen in Fig. 4.5.

In Fig. 4.6, 4.7 and 4.8 we can see how the interface to the system's functions leans heavily on natural language elements. The design ideal was for the system to be as accessible and the least intimidating possibile, in order to let the user extract the maximum amount of information in an easy way. Not only the terms but also the phrasing were deemed to be important: every interaction is broken down into what are though to be manageable steps. For example, a user

Algorithm 10 Mutual information algorithm

```
1: X parent variable in the BN DAG
2: Y child variable in the BN DAG
3: E set of current evidence in the BN
4: if X ∈ E then
5:     return -1
6: end if
7: joint = p_XY(X, Y|E = e)
8: Y_marginal = marginalise joint over X
9: X_marginal = marginalise joint over Y
10: mutual_information = 0
11: for y in Y_marginal do
12:     for x in X_marginal do
13:         j =entry in joint corresponding to y and x
14:         if j is 0 then
15:             mutual_information+ = 0
16:         else
17:             mutual_information = j ∗ log(j/(y ∗ x))
18:         end if
19:     end for
20: end for
21: return mutual_information
```

isn't asked to provide all evidence at once but is asked for one at a time, when necessary. The idea is to reduce cognitive overload on the expert's part so that he may be able to better concentrate on the task at hand, not on the tool he is using to achieve it.

Another important aspect is to translate the information into a language that would make semantic sense to the user; that is it should be part of the user's *ontology*, the set of concepts part of his World. This can be seen in Fig. 4.6 where the notion of d-separation in the underlying BN's DAG is remapped to a higher-level concept that is part of the user's ontology, like that of variables affecting each other's values or not. The underlying theme of the work carried out in this thesis is to translate *information* present in a data set into *knowledge* for a medical expert user. This is done in various ways, for example through the use of dialogue, to bring the data to a form that is manageable and comprehensible.

A final element is that of a coherent use of colours, even in the experimental command-line interface that has currently been developed. In the generated phrases the source variables are always highlighted in magenta, the evidences in green and the results of the queries in cyan. It is hoped that this guide the user's eye towards the important elements of what he had set out to achieve, thus anchoring his experience and letting the marginal elements fade into background over time.

The grammars for the generation of the natural language are now presented, defined in Extended Backus-Naur form (EBNF).

One of the most important parts of this work is the validation of this interface with real medical expert users, the results of which are presented in chap. 5.

Algorithm 11 Grammar generating dialogue output

1: Next_state := 'var 1' | ... | 'var n'
2: Value := 'val 1' | ... | 'val k'
3: 'highly unlikely' | 'very unlikely' | 'unlikely' | 'not plausibly' | 'plausibly' | 'possibly' | 'likely' | 'very likely' | 'highly likely' | 'certain'
4: Output := 'The next most probable state is ' Probability Next_state 'Enter to accept or n to refuse: ' (Still_to_explain ',')* | 'Is state ' Next_state ' with value ' Value ' more correct?'
5: Still_to_explain := 'var 1' | ... | 'var n'

Algorithm 12 Grammar generating independencies query output

1: Source := #user input#
2: Evidence := 'var 1' | ... | 'var n'
3: Separated := 'var 1' | ... | 'var n'
4: Output := 'Given source variable' Source 'and given evidence: ' (Evidence ',')* 'the following variables have no effect: ' (Evidence ',')*

Algorithm 13 Grammar generating conditional probability query output

1: Source := #user input#
2: Evidence := 'var 1' | ... | 'var n'
3: Output := 'Given target variable' Source 'and observed evidence: ' (Evidence 'with value: ' Value)* 'then the predicted values for ' Source 'are: ' (Evidence 'with value: ' Value)*

Algorithm 14 Grammar generating MPE query output

1: Source := #user input#
2: Evidence := 'var 1' | ... | 'var n'
3: Value := 'val 1' | ... | 'val k'
4: Output := 'Given observed evidence:' (Evidence 'with value: ' Value)* 'the most probable configuration of the other variables is: ' (Evidence 'with value: ' Value)*

Table 4.4. Probability quantifiers in natural language

Probability range	Natural language quantifier
(0, 0.2)	"highly unlikely"
(0.2, 0.3)	"very unlikely"
(0.3, 0.4)	"unlikely"
(0.4, 0.5)	"not plausibly"
(0.5, 0.6)	"plausibly"
(0.6, 0.7)	"possibly"
(0.7, 0.8)	"likely"
(0.8, 0.9)	"very likely"
(0.9, 1)	"highly likely"
(1)	"certain"

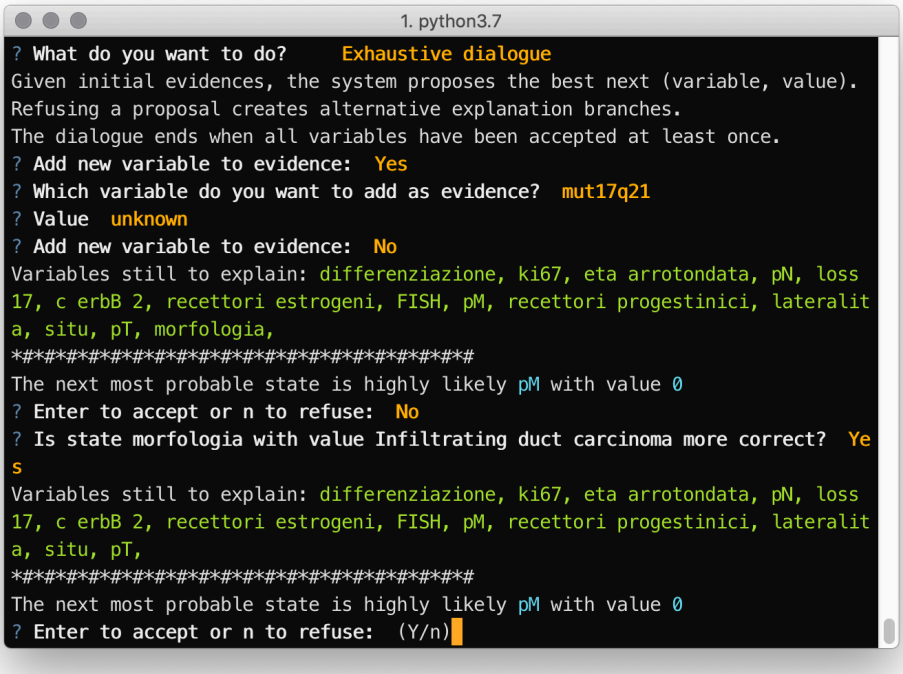


Figure 4.5. Interface while executing a Dialogue

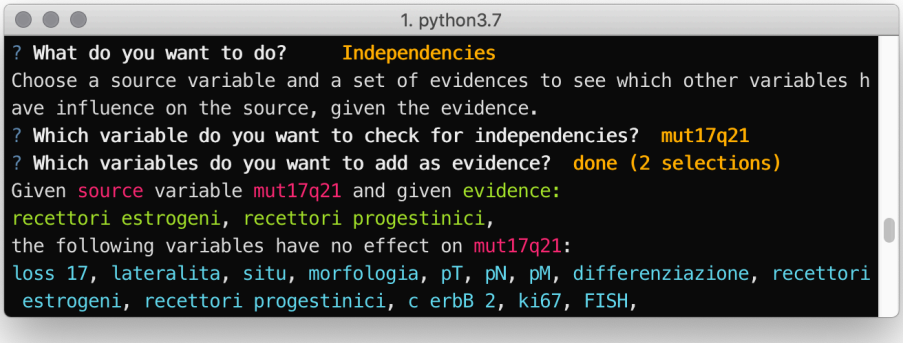


Figure 4.6. Interface while executing a query on the d-separations

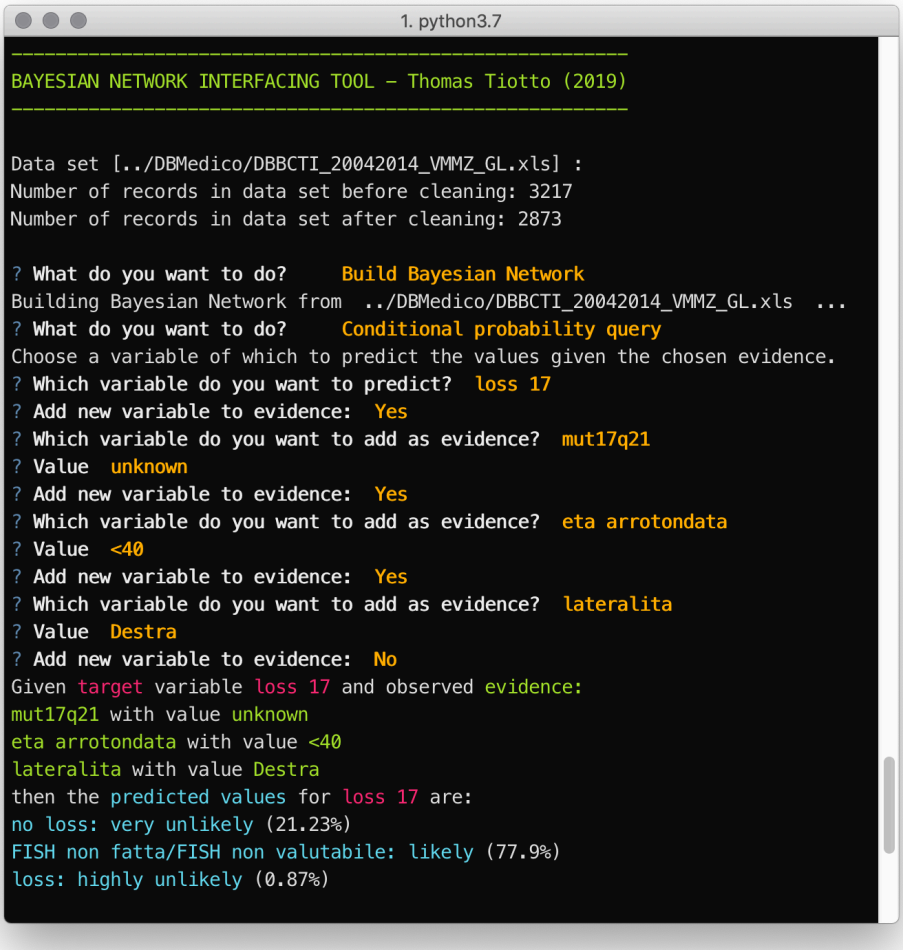


Figure 4.7. Interface while executing a conditional probability query

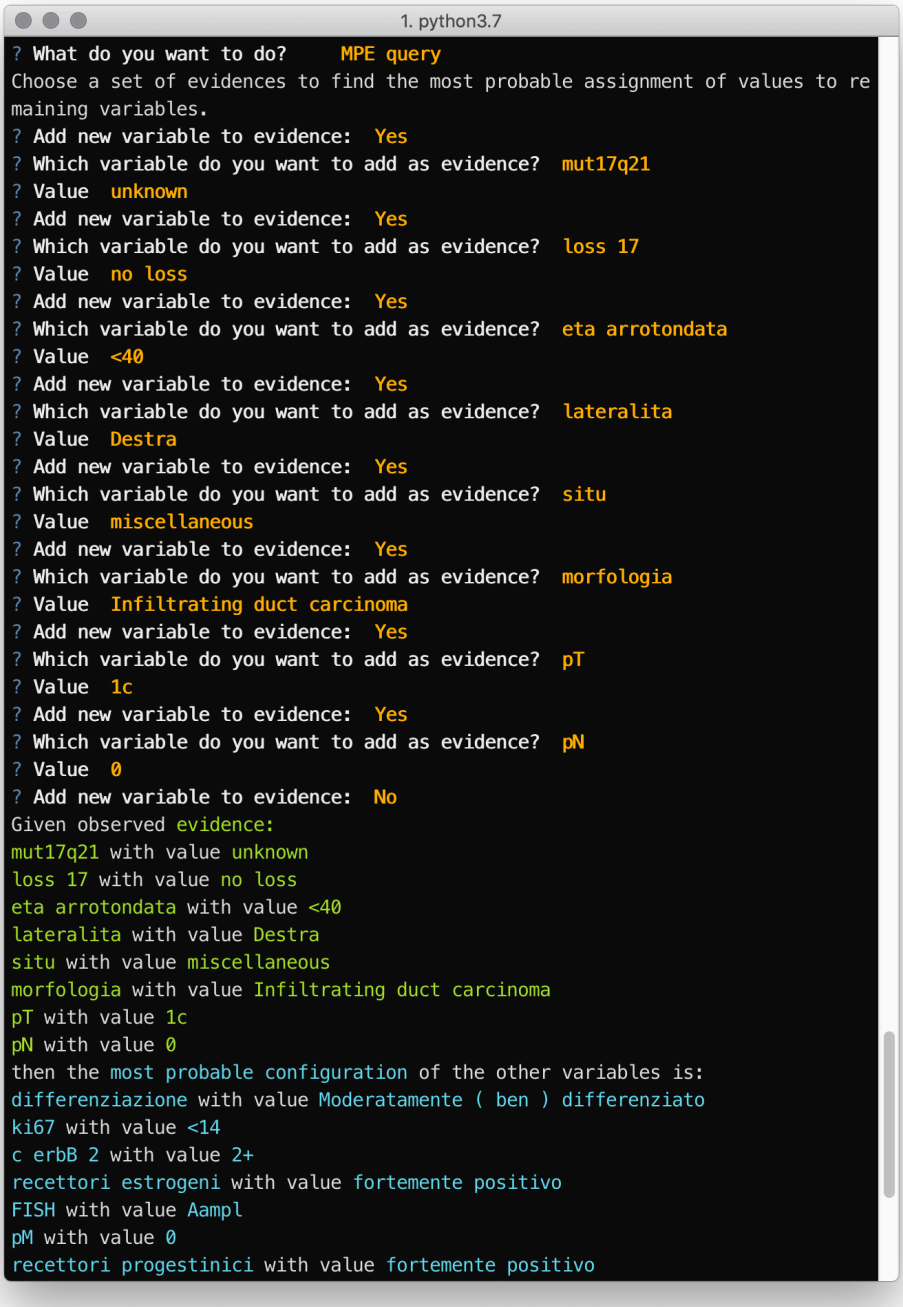


Figure 4.8. Interface while executing an MPE query

4.5 Validation

discutere con Vittoria come validare al meglio

Having direct access to expert pathologists has not only helped in guiding research into the theoretical explainability properties of the system but also enabled their validation. There are two main validation streams to be addressed: from the clinical point of view and from the explainability one, with the results of the latter depending on those of the former.

4.5.1 Clinical validation

A validation of the methods carried out in this thesis in their adherence to established clinical literature, is of paramount importance. A failure on the Bayesian Network's part in capturing the true relationships between the variables would hamper it in being able to give any meaningful representation of them. For the expert to even start to trust the system or to be able to make sense of its outputs, it is vital that there be as little cognitive dissonance between the experts' basic beliefs and expectations and those that he sees represented in the system.

For this reason, the initial validation phase with the Istituto Cantonale di Patologia concentrated on the clinical validation aspect. The methodology chosen to clinically validate the system was for a representative of the ICT to formulate a series of natural language queries; each one of these was annotated with the specific queried variable in the network and its value together with the known values of other variables. The ICT's delegate included the expected reply to the queries together with its probability, based on the latest medical literature and their personal expertise. So, each query asked for the probability truth value of the following:

$$(var_1 = a \wedge \ldots \wedge var_n = z) \Rightarrow var_k = k \quad var_1 \ldots var_n \in V, var_k \in V \setminus \{var_1 \ldots var_n\} \quad (4.2)$$

This would translate into natural language into: "If var_1 is a and ... and var_n is z, how probable is it that var_k k?". The complete series of eight questions, together with the expected values, is shown in Tab. **??**. It is quite evident that all these validation questions are instances of the Bayesian Network Updating problem, in particular they are Conditional Probability Queries (as defined in 3.5.2).

inserire tabella qui o in Results

4.5.2 Explainability validation

Having validated the system in terms of its adherence to clinical literature, it could then also meaningfully be validated from an explainability point of view. The main question to be addressed is its capacity to relate to the expert user. Is the system able to engender the user's trust? In doing so, is she able to extract more knowledge from existing data when using the system than not? Especially in cases where there may be a dearth of data, can the expert maximise the benefit from the available information? Does the user subjectively feel that the system may positively impact her work? These are all hard questions to answer, as there is a very high degree of subjectivity involved. Thus to attempt to answer them, the chosen methods were borrowed from the social sciences.

In an earlier stage, the experts were introduced to the system in prototype form and instructed on the possible use cases it offered. Then, they were asked to keep track of times they felt they could have used the system, had it been available, during their day-to-day work and to report back to me with the query they would have submitted to the system. If the query were in

the realm of possibilities offered by my program, I would submit the outputs back to the users together with the following questionnaire:

definire questionario da mandare con le domande

This process would both give feedback on the performance of the system and also inform its design.

In a later phase, having finalised the system's design, I provided it to the experts at the ICT for their use in their daily work. After **xx time** I followed up with them with an interview based on the following format:

definire formato e metriche intervista

This enabled me to quantify the performance of the system, as perceived by its users in a real setting, over an extended period of time.

4.6 Summary

This chapter has presented

Chapter 5

Results

5.1 Introduction

per ogni sezione chiedere a Vittoria e Gine come li userebbero in clinica

5.2 Implemented Tool

5.2.1 Overview

The system often referenced and shown in some detail in Sec. 4.4 is a proof-of-concept terminal-based tool that was developed in order to test the hypotheses laid out in Chap. 1. The defining motivation was to have a working software that could be given to a number of clinicians at the Istituto Cantonale di Patologia (see Subsec. 4.2.1) in order to have them test the theoretical questions, defined in 1.3, that are at the core of this thesis. Being this only a prototype, the implementation was carried out using Python 3.6, as this was the language that let me best focus on rapid development, due to my familiarity with it and to its vast array of available libraries. The strengths of Python, no compiler

Despite never meaning to be a production software, particular care was taken in the design of the interface, as described in Subsec. 4.4.2, in line with the spirit of this work that is to study human-machine interaction.

In the following, the various interaction modalities with the tool are presented using screen-shots. The basic methods underlying the tool have already been discussed at length in Sec. 4.4 so the current examination will focus on the interface, as perceived by the user, and how these methods have been incorporated into the system. Where relevant, the information and descriptions given in Sec. 4.4 will be integrated.

Fig. 5.1 shows the first screen presented during use. The user has the possibility of inputting the path to the dataset to use, or to accept the hardcoded one, that in this case is the one described in Sec. 4.2. Next, the number of entries before and after preprocessing are shown; the data set in question see its number of valid records go from 3217 to 2873, after the rules of Tab. 4.3 have been applied. The “Inspect data set” and “ML” options are only for testing and validation purposes; the former surfaces a pair of options to visualise the distribution of the data set’s variables’ values and the normalised entropies, the latter runs the Machine Learning tests that will be discussed in Sec. 5.3. The main focus will be the “Build Bayesian Network”

section as this is where all the methods discussed in Chap. 4 are to be found. Selecting this option automatically uses pomegranate to construct a Bayesian Network model of the selected data set. The user is then shown what is the main menu of the application, as can be seen in Fig. 5.2.

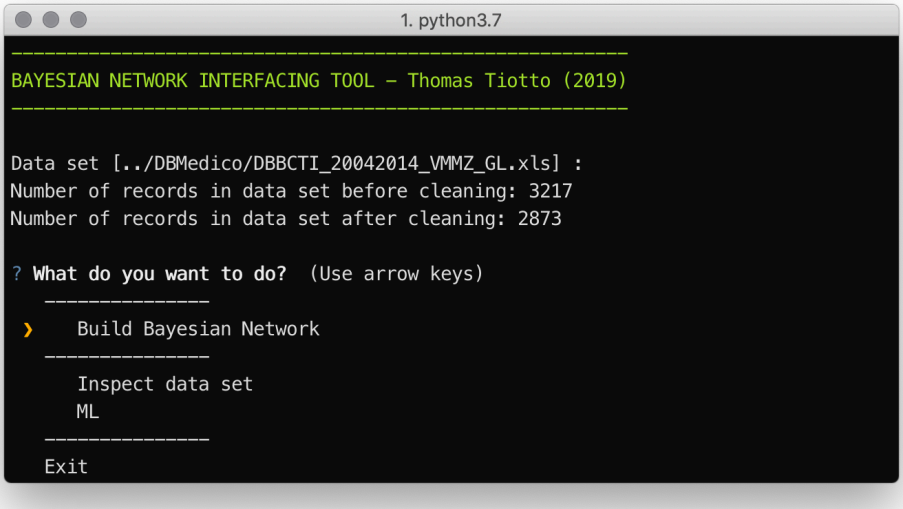


Figure 5.1. Initial screen in the developed tool.

5.2.2 Independencies

The "Independencies" interaction mode gives the expert the possibility of verifying which d-separations (defined in 3.4.3) exists in the constructed Bayesian Network's DAG (defined in 3.4.1) . The concept of d-separation is here reworded into a higher-level notion of "choosing a source variable and a set of evidences to see which other variables have influence on the source, given the evidence". This is because Dr. Vittoria Martin, my main contact at the Istituto Cantonale di Patologia, initially had difficulty in conceptualising at the level of graph theory. For this same reason, after having chosen first the source variable and then the observed set of evidence variables, the user is presented with an output both in graph (Fig. 5.4) and in natural language (Fig. 5.3) form. Having both output forms was seen to reduce the confusion associated with such an unfamiliar concept to users trained in the medical sciences.

From an explainability point of view it was seen that ...

In clinical practice ...

5.2.3 Conditional Probability Query

Conditional probability queries (defined in Subsec. 3.2.3) were seen to be instinctively understood by the clinicians at the ICT. Indeed, many of the natural language questions that were defined to validate the system (see Sec. 4.5) were naturally instances of this type of query.

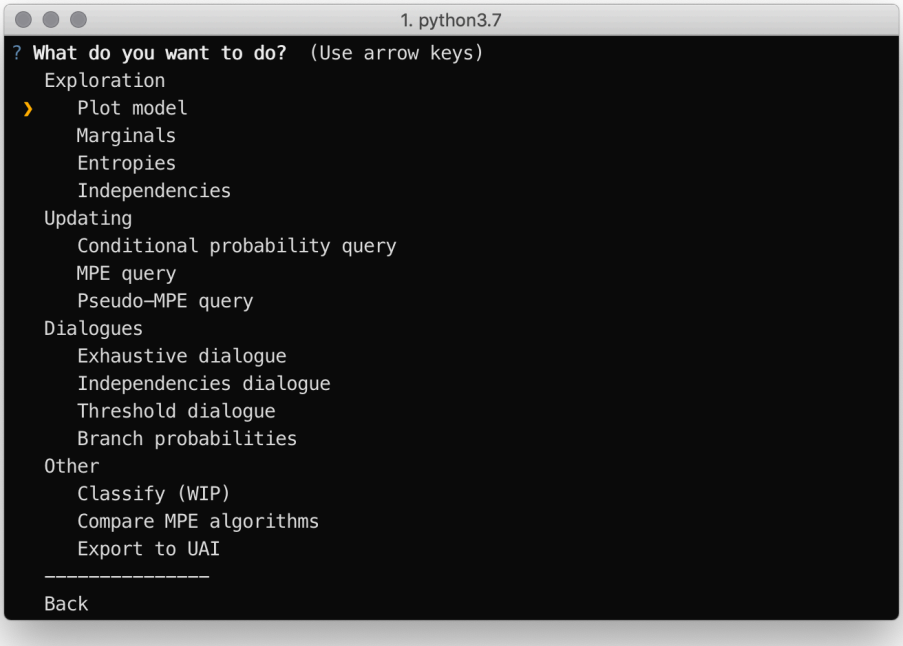


Figure 5.2. Main interaction menu.

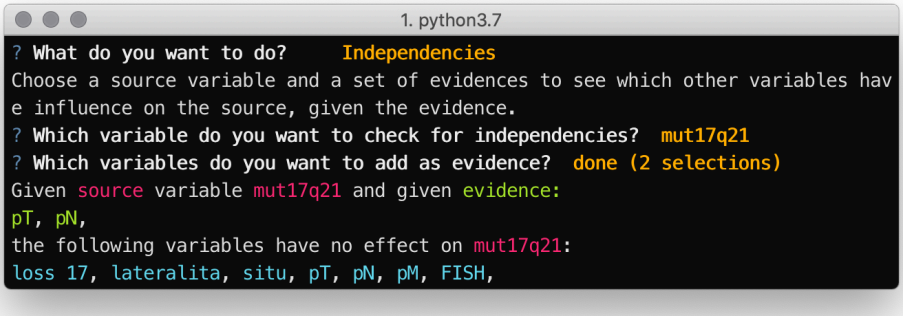


Figure 5.3. Independencies query natural language output.

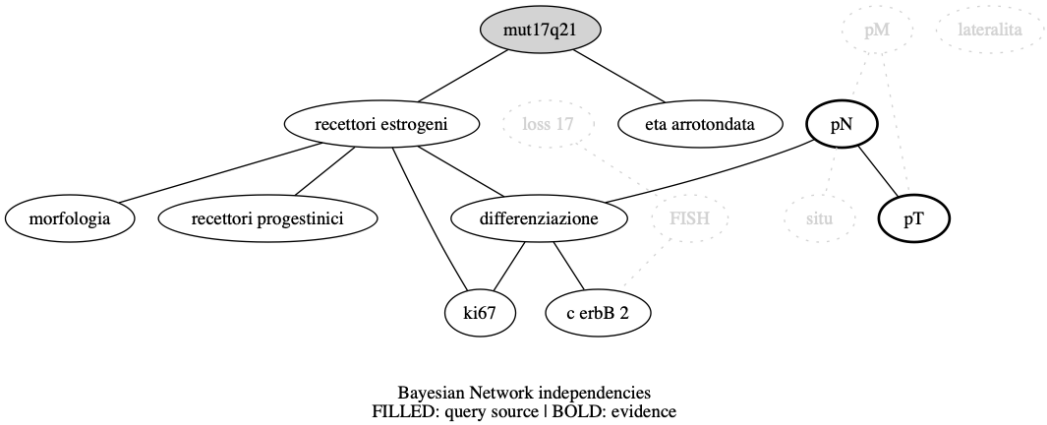


Figure 5.4. Independencies query graph output.

The user is asked for a target variable of which to observe the conditioned values and for a set of variables, together with their observed values. The output, in natural language (Fig. 5.5), includes all elements of the query in a colour-coded manner. The answer to the question posed (in cyan) shows the probability of each of the states of the target variable, quantified in natural language, using the coding defined in Tab. 4.4, and as raw probabilities, shown as a percentage.

From an explainability point of view it was seen that ...

In clinical practice ...

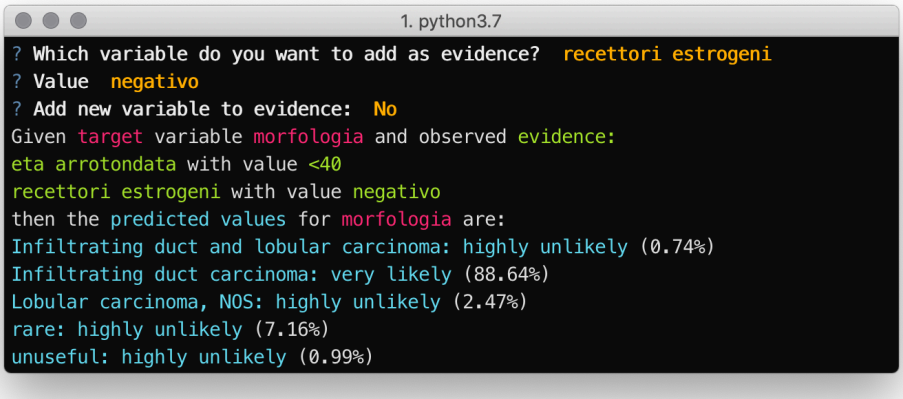


Figure 5.5. Conditional probability query output.

5.2.4 MPE Query

Queries of the MPE type (defined in Subsec. 3.5.2) were also understood naturally when a bridge to the clinical practice had been established. When presented at an abstract, mathematical level, the experts of the ICT weren't clear of the utility of such a query class. It was soon understood that an MPE query corresponds to a familiar concept to any clinician, that of "a maximally likely patient profile". That is, given a set of known parameters, it is of interest for the expert to find which is the most likely assignment to the other. As each record in the data set represent a patient's clinical profile, this is equivalent to finding the most probable patient given a set of know values.

Another way than an MPE query makes clinical sense, is in the crucial task of predicting missing values for a patient. This is not an unlikely case, as discussed in Subsec. 4.2.2, because there is more than one reason that a patient may be missing one or more entries in his clinical profile. Executing an MPE query with the known patient's values will yield the most probable assignments to the missing ones and is thus equivalent to a prediction task. The clinical significance of such an interaction mode can also be inferred from the fact that many of the natural language questions that were defined to validate the system (see Sec. 4.5) were seen to map onto instances of this type of query.

At a technical level, the MPE calculation is executed using pgmpy's `map_query` function. The output, shown in Fig. 5.6, presents in colour-coded natural language the input evidence and most probable assignments to the remaining variables.

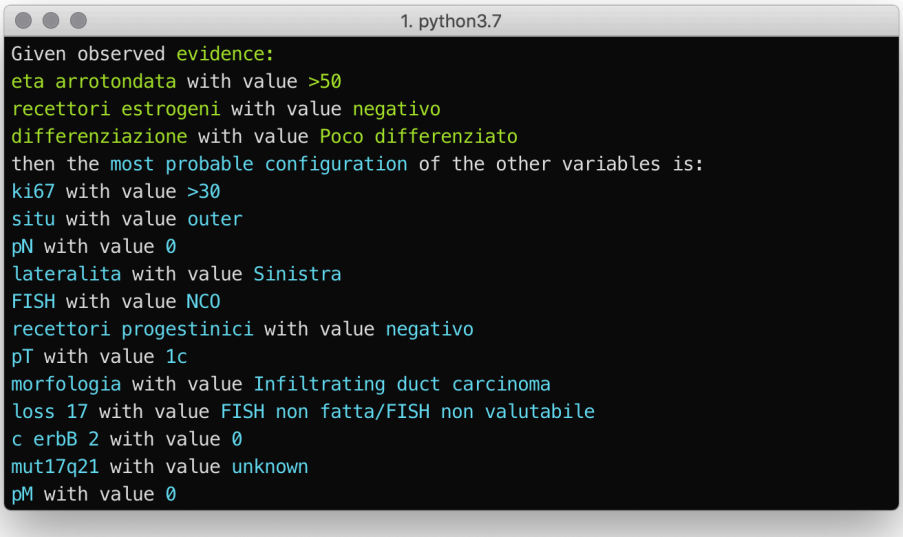


Figure 5.6. MPE query output.

5.2.5 Pseudo-MPE Query

The "Pseudo-MPE query" interaction mode is aimed at generating a *max over marginals* assignment using the methods described in Subsec. 4.4.2 under the **Pseudo-MPE from Initial Evidence** header. Note that if the cardinality of the set of variables to explain is one, i.e. $|E| = |V| - 1$, with E the evidence set and V the set of variables in the BN, then the Pseudo-MPE and true MPE assignments will be identical.

The user is first asked for the probability threshold under which to discard the $(state, value)$ pairs whose probability is deemed too low. Then, after being asked for the initial observed evidence, the expert is presented with the constructed polytree (defined in Subsec. 3.4.2), an example of which can be seen in Fig. 5.7. This polytree will have the initial evidences, that the expert specified, as roots and a single chain of $(state, value)$ pairs, each one being quantified with its probability (in natural language) given all of its ancestors.

From an explainability point of view it was seen that ...

In clinical setting ...

5.2.6 Exhaustive Dialogue

The three "dialogue" are the most experimental interaction modes and thus also the most alien to an expert user. None of the natural language questions defined by the ICT in form form described in 4.5.1 could be directly mapped onto such a dialogical process. Because of the novel nature of such a knowledge-extraction process, three different versions were implemented, with each one adding a different set of behaviours, compared to the Exhaustive version described in this subsection. This let me better explore the space of possibilities and understand which features were preferred by the clinicians of the ICT, both as a means for knowledge-extraction from the data set and from a comprehensibility point of view. It should here be noted that comprehensibility of the outputs is a *necessary* but not sufficient condition to be able to gain knowledge from data. Both the variants, the independencies-aware one and the thresholded one, aim to prune the space of variables proposed to the user, in order to reduce her cognitive load.

The Exhaustive Dialogue, as described in much more detail in Subsec. 4.4.2 under the **Dialogues** header, is so named because it ends only when the expert user has reviewed all the variables present in the data set. It starts by asking the clinician for a set of initial evidences and from thereon-after iteratively proposes the least entropic $(state, value)$ pair, based on the accumulated evidence (the rationale behind this is explained in Subsec. 4.4.1). An example of such an ongoing interaction is shown in Fig. 5.8.

From an explainability point of view it was seen that ...

In clinical setting ...

5.2.7 Independencies Dialogue

The first variant to the Exhaustive Dialogue takes the approach of excluding variables based on their d-separations (defined in Subsec. 3.4.3) in the underlying DAG (defined in Subsec. 3.4.1). Thus, in the general case, the cardinality of set of variables proposed to the user varies in a non-linear way. In the Exhaustive Dialogue, presented under the previous header, the relationship between the set of variables still to explain at step t $W_t = V \setminus E_t$, with V all the variables and

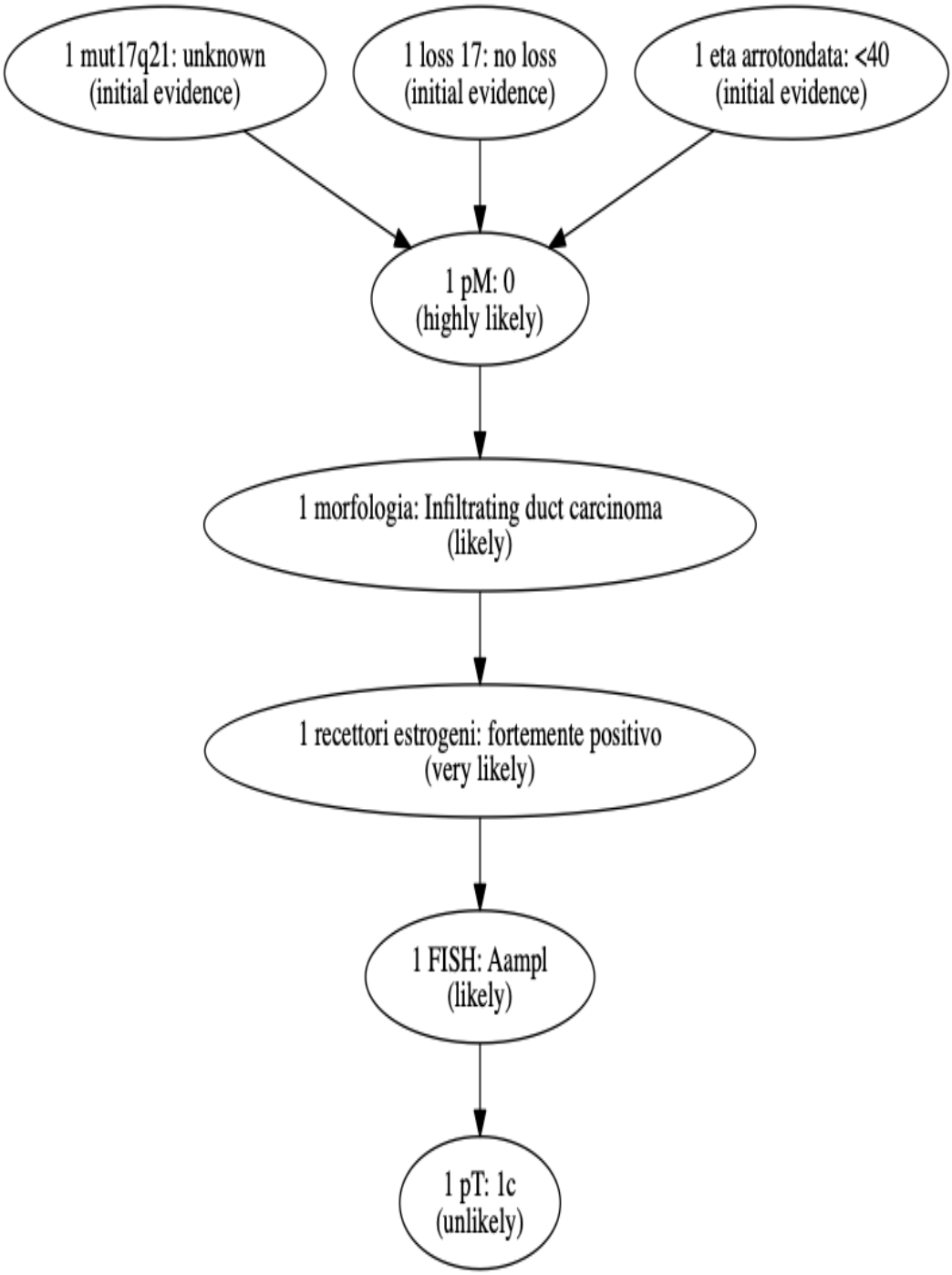


Figure 5.7. Pseudo-MPE query output with threshold 0.5.

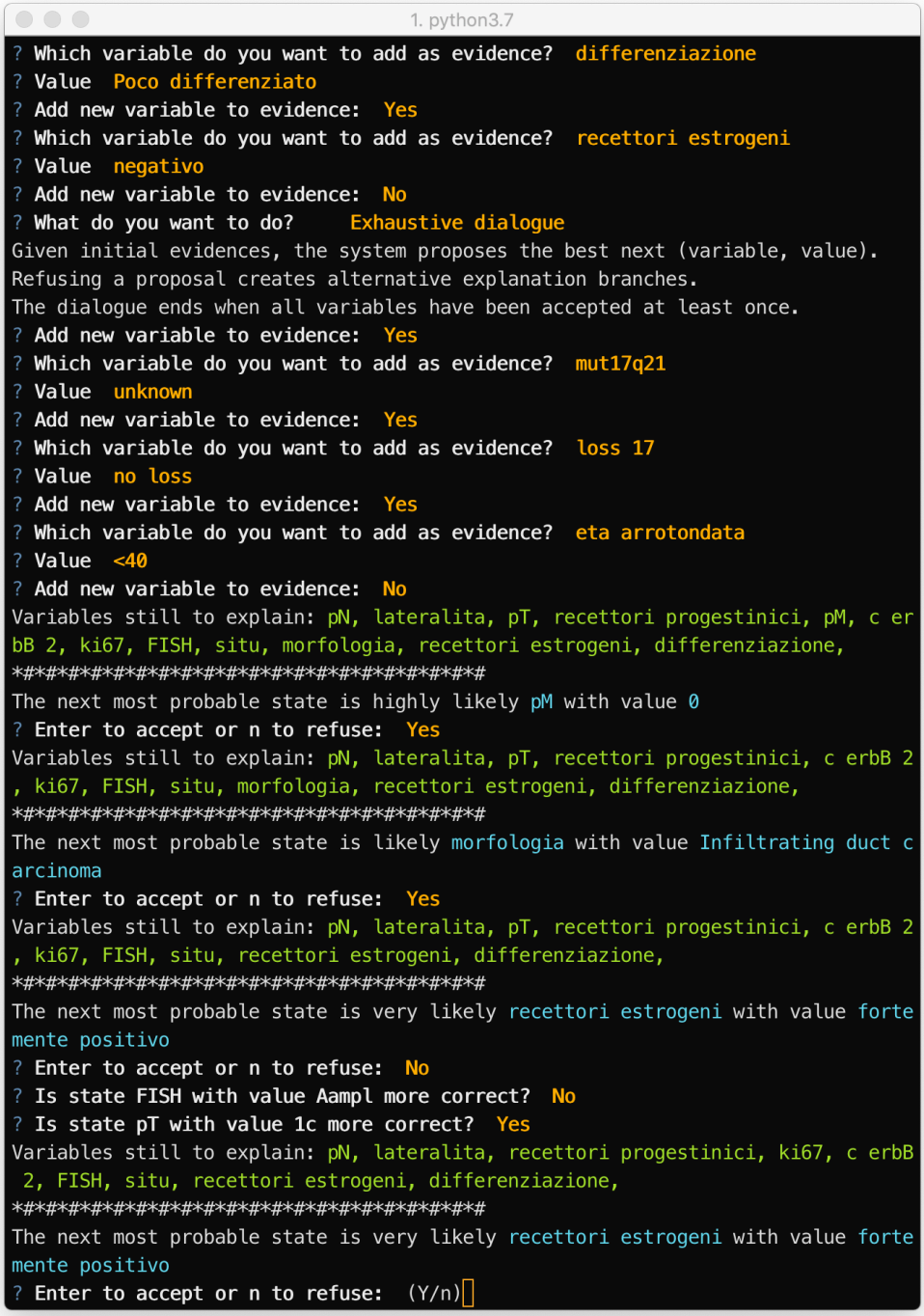


Figure 5.8. Ongoing Exhaustive Dialogue.

E_t those already added to evidence, obeyed the recurrence relation:

$$W_0 := V \tag{5.1}$$
$$E_0 := \emptyset \tag{5.2}$$
$$|W_{t+1}| = |W_t| - 1 \tag{5.3}$$
$$|E_{t+1}| = |E_t| + 1 \tag{5.4}$$

That is, at each step t of the Exhaustive Dialogue, one variable moves from the set still to explain W to the explained one E. In the Independecies Dialogue, this relationship depends on the set of variables Z that are d-separated from those already in E. The relationship between the cardinalities is modelled by an operator ζ that is unique to the DAG of the BN (or to any *I-equivalent* one):

$$W_0 := V \tag{5.5}$$
$$E_0 := \emptyset \tag{5.6}$$
$$|W_{t+1}| = \zeta(|E_t|) \tag{5.7}$$
$$|E_{t+1}| = |E_t| + 1 \tag{5.8}$$

As d-separation is not monotonic (adding a variable to E may open new paths), the cardinality of the set W may vary, from the point of view of the user, in an unpredictable manner. The user is supported by an updated view of the independencies in the graph (an example during the dialogue is shown in Fig. 5.9.

Before receiving feedback from the ICT, the visualisation of the independencies was the one shown in Fig. 5.10. The most striking difference was the use of colour-coding to identify the role and the separateness of variables with pink identifying the query variables, blue the evidences, red the separated variables and green the connected ones. As already noted in Subsec. 5.2.2, the concept of d-separation turned out to be quite unfamiliar to the clinicians of the ICT so the first priority was to represent the concept visually in the clearest way possible. This was achieved, and confirmed in its efficacy by the ICT, by fading the separated variables and marking those in evidence in bold, as can be seen in Fig. 5.9.

A second point of confusion, that hadn't been foreseen, was the use of directed arcs to represent the DAG. In the visual representation of a Bayesian Network, an arc between two variables represents a correlation between their values while the direction identifies the $parent$ and the $child$ in the relationship; for example the graphical representation $X \rightarrow Y$ means that X is the parent of Y. This is crucial because the fundamental idea of a BN model is to factorise the joint distribution such that each variable's values depend only on that of its parents; the concept of Conditional Probability Table is explained in Sec. 3.5 and a some more examples can be seen in Subsec. 4.3.2 under the **MPE** header. Nonetheless, Dr. Vittoria Martin explained that the DAG representing a BN is very similar to diagrams used during clinical research, with the crucial difference that in those a directed arrow represents *causation* and not *correlation*. A correlative relationship would have usually been represented by an undirected edge. For this reason, I elected to substitute the DAG representation of the BN with that of an undirected graph. The ICT representative confirmed that this aligned much better with her intuition.

The third element of difference, is the addition of the Mutual Information coefficient (defined in Subsec. 3.3.3) on the arcs connecting each couple of variables; the coefficient also scales the width of its associated edge, giving further visual feedback to the user. This was a direct request from the ICT's representatives as they felt that this would increase their understanding

of the relationships between the variables. D-separation is binary while Mutual Information can give the practitioner a much greater (theoretically, infinite) range of information. For example looking at Fig. 5.9 it is quite easy to see that, while “morfologia” and “recettori estrogeni” are d-connected, the amount to which they influence each other’s values is small, compared to other connected variables. Some arcs, for example those belonging to “mut17q21”, are missing the Mutual Information coefficient because either the parent or the child variable is in the evidence set.

From an explainability point of view it was seen that ...

In clinical setting ..

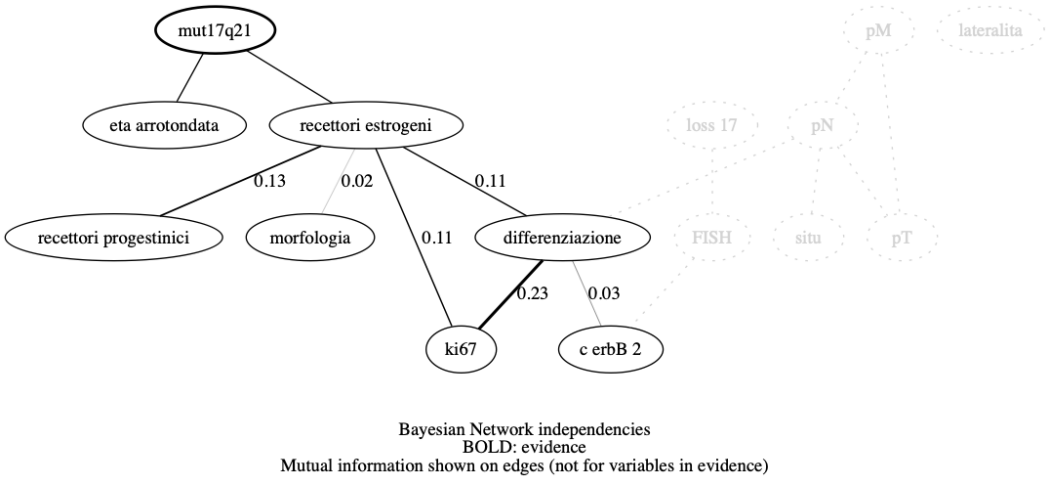


Figure 5.9. Ongoing Independencies Dialogue.

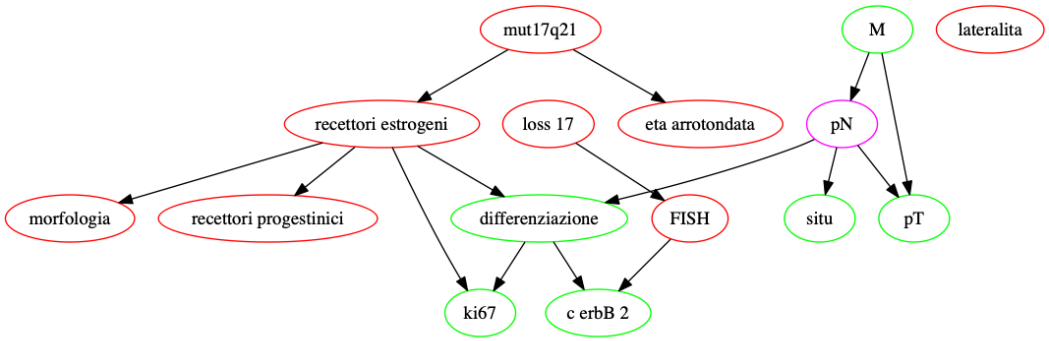


Figure 5.10. Old visualisation during Independencies Dialogue.

5.2.8 Threshold Dialogue

The final Dialogue variant adopts a different strategy for pruning, namely one based on the probability of the proposed tuples and on the number of times that they have been refused by the expert. The cardinality of the set W of states to explain decreases linearly, like in the Exhaustive Dialogue, but potentially with a slope coefficient $\alpha \leq -1$, as many states may be infra-threshold i.e. too unprobable to be considered. Unlike the Independencies Dialogue, the cardinality of W can't increment:

$$W_0 := V \tag{5.9}$$
$$E_0 := \emptyset \tag{5.10}$$
$$|W_{t+1}| = \alpha |W_t| \tag{5.11}$$
$$|E_{t+1}| = |E_t| + 1 \tag{5.12}$$

The default values for the threshold and the maximum number of times a $(state, value)$ tuple could be proposed, were decided together with the ICT and set to xx and xx, respectively.

From an explainability point of view it was seen that ...

In clinical setting ..

5.2.9 Combination of Features from All Dialogue Variants

From an explainability point of view it was seen that ...

In clinical setting ..

5.3 Bayesian Networks Predictions Strength

5.4 Validation results

5.4.1 Clinical Validation

5.4.2 Explainability Validation

5.5 Pseudo-MPE Evaluation

This is not a user-facing feature *per se* but a way of running a test to compare the outputs of the Pseudo-MPE algorithm with the exact solution (this is described in detail in Subsec. 4.4.2 under the **MPE Algorithms Comparison** header. I also implemented the possibility of scoring the MPE calculated with pgmpy's `map_query` and with DAOOPT. The Hamming and Jaccard distances between these should have been zero, as they both use exact methods to solve the MPE problem. It was seen that this was not the case and this lead to the discovery of the problems described in Sec. 5.7. As is explained in Sec. 5.7, the benchmark against which to compare the Pseudo-MPE was taken to be pgmpy's `map_query` function.

The tests were run ...

5.6 Pseudo-MPE Complexity

5.7 Issues

5.7.1 Zero Probabilities in Learned CPTs

It was noticed that some of the Conditional Probability Tables (CPT/CPD), that pomegranate learned from the data set (i.e. by solving the problem defined in Subsec. 3.5.1), presented entries with 0 value.

The assignment of the extreme probabilities 1 and 0, while perfectly coherent in the frequentist approach, is not in line with the Bayesian one. This is because of the different conception of probabilities between the two approaches, as discussed in Subsec. 3.2.1. A frequentist perspective would happily assign probability zero to an event not present in the data set while a Bayesian would refrain to do so, as having a zero or one prior belief makes every posterior, calculated using Bayes' Rule (Eq. 3.4), also zero or one. The necessity to avoid assigning prior probability beliefs of 0 or 1, has been named "Cromwell's Rule" by Jackman [2009].

It would be an epistemological error to be absolutely sure, one way or another, of any *a posteriori/syntetic proposition* and thus we should refrain from assigning absolute belief or disbelief into their truth value. This, apart from questions regarding the ontological determinism of Reality, is also against the intended use of statistics, that should be applied in cases of uncertainty and not in those of deterministic mechanism. We should do so only for *a priori propositions*, for example those referring to Mathematic or Logic, but not for any *a posteriori statements* i.e. those whose truth value is based on experience (for a definition of these see the Introduction of Kant [1998]).

To solve the issue in the context of my work, I elected to apply a post-learning correction inspired by *Laplace Smoothing*; specifically I added a small positive constant to every zero-valued entry in the CPTs in pomegranate's model. The algorithm is based on adding and subtracting "probability atoms" ϵ with the objective of removing zeros in the CPTs and maintaining the normalisation, so that the result is still a valid probability distribution (for an introduction, see Subsec. 3.2.1). We work with probability atoms ϵ so as not to incur in numerical imprecisions in the implementation, as only additions and subtractions are used. Every zero-valued element in a CPT column has a quantity $\epsilon \times \#!0$ added to it, with $\#!0$ the number of elements in the distribution that are not zero, and every non-zero entry has $\#0$, the number of zero elements, atoms subtracted from it. The end result is obviously still a correctly normalised probability distribution because given a probability distribution P, with n elements p, of which $\#0$ are zero, and the distribution P^* resulting from to the described procedure:

$$\begin{aligned} & \sum_{i=0}^{n} p_i = 1 \\ \wedge \quad & 1 - \{\#0 \times [(n - \#0) \times \epsilon]\} + \{(n - \#0) \times [\#0 \times \epsilon]\} = 1 \\ \Rightarrow \quad & \sum_{i=0}^{n} p_i^* = 1 \end{aligned}$$

The pseudocode is shown in Alg. 15.

If the procedure were applied to the CPT reproduced in Tab. 5.4, it would yield the one shown in Tab. 5.1.

Algorithm 15 Epsilon Smoothing algorithm pseudocode

```
1: ϵ = smallest positive constant
2: for s in model CPTs do
3:     for c in s's columns do                ▷ distributions of values are organised column-wise
4:         num_zeros = number of zero-valued entries in c
5:         num_non_zeros = |c| - num_zeros
6:         for v in c do
7:             if v equal to 0 then
8:                 v+ = ϵ × num_non_zeros
9:             else
10:                v− = ϵ × num_zeros
11:            end if
12:        end for
13:    end for
14: end for
```

Table 5.1. recettori estrogeni CPD

		mut17q21	
		0	1
rec. estr.	0	$0.68-\epsilon$	0.13
	1	$0.0+2\epsilon$	0.02
	2	$0.31-\epsilon$	0.84

5.7.2 MPE Calculation

The main issue encountered during the implementation of the system described in Sec. 5.2, was that of correctly calculating the MPE. Initially, an attempt was made to write a custom function `export_model_to_uai` to generate a UAI file (described in 4.3.1) directly from the pomegranate Bayesian Network model. This UAI file was fed to DAOOPT to generate the MPE solution as recounted in Subsec. 4.3.2 under the **MPE Algorithms Comparison** header. When compared with the MPE solution generated directly using pgmpy's `map_query` function, it was seen that these disagreed in almost all cases. This shouldn't have been the case as both were generated using exact methods: Variable Elimination in pgmpy's case and AND/OR Branch-and-Bound for DAOOPT.

While investigating the cause for this divergence, I came across an undocumented feature of pgmpy: a `UAIWriter` class that should have converted the pgmpy-based model (that was converted from the pomegranate-based model as recounted in Subsec. 4.3.2 at the **Pairwise Correlations** header) to the correct UAI file representing it. When this alternative UAI file was used as input for DAOOPT, the resulting MPE not only diverged from that calculated based on `export_model_to_uai`, that was to be expected, but also from that calculated directly with pgmpy using `map_query`, that was surprising.

The conversion from the pomegranate-based to the pgmpy-based model was thoroughly tested using conditional probability and independencies query, so the issue is most likely to be found elsewhere. As the DAOOPT MPE solution generated starting from the UAI differs from the one calculated directly with pgmpy, there must be a bug either in pgmpy's UAI exporter or in its inference method. It is unclear if the following is still an issue (`https://github.com/pgmpy/pgmpy/issues/856`) but a series of test on simple networks, where the MPE calculation were carried out manually, seemed to confirm that `map_query` was returning the correct MPE solution.

For example, in the small BN whose structure is shown in Fig. 5.11 and the CPDs of the nodes in Tab. 5.2, 5.3, 5.4 and 5.5, pgmpy's `map_query` and DAOOPT returned different solutions to the following MPE query:

$$\begin{aligned} argmax_{x,y,z}\mathbb{P}(&\text{differenziazione} = x,\\ &\text{mut17q21} = y,\\ &\text{recettori estrogeni} = z|\text{eta arrotondata} = 0) \end{aligned}$$

`map_query` returned the assignment:

$$\text{differenziazione} = 1, \text{mut17q21} = 1, \text{recettori estrogeni} = 2 \tag{5.13}$$

while DAOOPT returned:

$$\text{differenziazione} = 0, \text{mut17q21} = 1, \text{recettori estrogeni} = 2 \tag{5.14}$$

In such a small network it is easy to verify that the probability of 5.13 is: $0.99 \times 0.84 \times 0.60 = 0.50$ and that it is the MPE solution; while that of 5.14 is $0.99 \times 0.84 \times 0.21 = 0.17$ that makes it obviously incorrect.

5.8 Summary

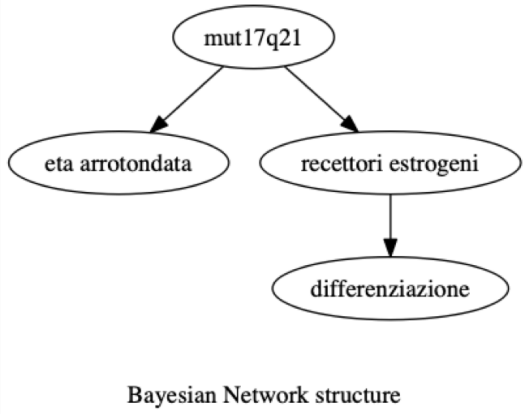


Figure 5.11. Independencies query natural language output.

Table 5.2. mut17q21 CPD

	0	1
mut17q21	0.006	0.99

Table 5.3. eta arrotondata CPD

		mut17q21	
		0	1
	0	0.42	0.04
eta arr.	1	0.42	0.17
	2	0.15	0.78

Table 5.4. recettori estrogeni CPD

		mut17q21	
		0	1
	0	0.68	0.13
rec. estr.	1	0.0	0.02
	2	0.31	0.84

Table 5.5. differenziazione CPD

		recettori estr.		
		0	1	2
	0	0.012	0.16	0.21
diff.	1	0.18	0.43	0.60
	2	0.80	0.40	0.18

Chapter 6

Conclusions

6.1 Reply to Introduction

6.2 Future developments

incorporate user feedback into the data set
Interacting meaningfully with machine lear
systems: Three experiments

Glossary

Bibliography

DAOOPT. URL https://github.com/lotten/daoopt.

Graphviz. URL https://www.graphviz.org.

Gurobi. URL https://www.gurobi.com.

IDSIA. URL http://www.idsia.ch.

NetworkX. URL https://networkx.github.io.

NumPy. URL http://numpy.org.

Pandas. URL https://pandas.pydata.org/about.html.

Gurobi. URL http://pgmpy.org.

scikit-learn. URL http://scikit-learn.github.io/stable.

Ashraf Abdul, Jo Vermeulen, Danding Wang, Brian Y Lim, and Mohan Kankanhalli. Trends and trajectories for explainable, accountable and intelligible systems: An hci research agenda. In *Proceedings of the 2018 CHI conference on human factors in computing systems*, page 582. ACM, 2018.

Paul Anand, Prasanta Pattanaik, and Clemens Puppe. *The handbook of rational and social choice*. Oxford University Press, 2009.

Or Biran and Kathleen McKeown. Human-centric justification of machine learning predictions. *IJCAI International Joint Conference on Artificial Intelligence*, pages 1461–1467, 2017. ISSN 10450823. URL http://www.cs.columbia.edu/{~}orb/papers/human{_}centric{_}justification{_}ijcai{_}2017.pdf.

Jeffrey S. Bowers and Colin J. Davis. Bayesian just-so stories in psychology and neuroscience. *Psychological Bulletin*, 138(3):389–414, 2012. ISSN 00332909. doi: 10.1037/a0026450.

Raphaela Butz, Arjen Hommersom, and Marko van Eekelen. Explaining the Most Probable Explanation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11142 LNAI, pages 50–63. 2018. ISBN 9783030004606. doi: 10.1007/978-3-030-00461-3_4. URL http://link.springer.com/10.1007/978-3-030-00461-3{_}4.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186, 2017.

Thomas Cover and Joy Thomas. *Elements of Information Theory*. 2006. URL http://staff.ustc.edu.cn/{~}cgong821/Wiley.Interscience.Elements.of.Information.Theory.Jul.2006.eBook-DDU.pdf.

Derek Doran, Sarah Schulz, and Tarek R. Besold. What does explainable AI really mean? A new conceptualization of perspectives. *CEUR Workshop Proceedings*, 2071, 2018. ISSN 16130073. doi: 10.2307/1541581.

Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

Lilian Edwards and Michael Veale. Enslaving the algorithm: From a âĂIJright to an explanationâĂİ to a âĂIJright to better decisionsâĂİ? *IEEE Security & Privacy*, 16(3):46–54, 2018.

Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pages 80–89. IEEE, 2018.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):93, 2018.

Max Henrion and Marek J Druzdzel. Qualtitative propagation and scenario-based scheme for exploiting probabilistic reasoning. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, pages 17–32. Elsevier Science Inc., 1990.

Denis J. Hilton. Conversational processes and causal explanation. *Psychological Bulletin*, 107 (1):65–81, 1990. ISSN 00332909. doi: 10.1037/0033-2909.107.1.65.

Istituto Cantonale di Patologia. Istituto Cantonale di Patologia. URL https://www4.ti.ch/dss/dsp/icp/chi-siamo/presentazione/.

Simon Jackman. *Bayesian Analysis for the Social Sciences*, page 609. 2009. ISBN 9780470011546.

Immanuel Kant. *Critique of Pure Reason*. The Cambridge Edition of the Works of Immanuel Kant. Cambridge University Press, New York, NY, 1998. Translated by Paul Guyer and Allen W. Wood.

Kalev Kask and Rina Dechter. Stochastic local search for Bayesian networks. *in Proceedings Seventh International Workshop on Artificial Intelligence and Statistics*, 1999. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.144.7172{&}rep=rep1{&}type=pdf.

Daphne Koller, Nir Friedman, Sašo Džeroski, Charles Sutton, Andrew McCallum, Avi Pfeffer, Pieter Abbeel, Ming-Fai Wong, David Heckerman, Chris Meek, et al. *Probabilistic Graphical Models Principles and Techniques*, pages 26–27. MIT press, 2009a.

Daphne Koller, Nir Friedman, Sašo Džeroski, Charles Sutton, Andrew McCallum, Avi Pfeffer, Pieter Abbeel, Ming-Fai Wong, David Heckerman, Chris Meek, et al. *Probabilistic Graphical Models Principles and Techniques*, pages 74–76. MIT press, 2009b.

Carmen Lacave and Francisco J Díez. A review of explanation methods for bayesian networks. *The Knowledge Engineering Review*, 17(2):107–127, 2002.

Zachary C. Lipton. The Mythos of Model Interpretability. (Whi), 2016. ISSN 00010782. doi: 10.1145/3233231. URL http://arxiv.org/abs/1606.03490.

P Lucas. Bayesian networks in medicine: a model-based approach to medical decision making. *Proceedings of the EUNITE workshop on Intelligent Systems in patient Care*, pages 73–97, 2001.

Radu Marinescu and Rina Dechter. AND/OR Search Spaces for Graphical Models. *Artificial Intelligence*, (171):73–106, 2006. ISSN 00043702. doi: 10.1016/j.artint.2006.11.003. URL https://www.ics.uci.edu/{~}dechter/publications/r126.pdf.

Vittoria Martin. Curriculum vitae. 5:13–15, 2012. URL https://m4.ti.ch/fileadmin/DSS/DSP/ICP/pdf/Collaboratori/VM{_}CV.pdf.

Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 2018.

Tim Miller, Piers Howe, and Liz Sonenberg. Explainable AI: Beware of Inmates Running the Asylum Or: How I Learnt to Stop Worrying and Love the Social and Behavioural Sciences. 2017. URL http://arxiv.org/abs/1712.00547.

Brent Mittelstadt, Chris Russell, and Sandra Wachter. Explaining explanations in ai. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 279–288. ACM, 2019.

Sharon Use Normand and David Tritchler. Parameter updating in a bayes network. *Journal of the American Statistical Association*, 87(420):1109–1115, 1992. ISSN 1537274X. doi: 10.1080/01621459.1992.10476266.

Judea Pearl and Rina Dechter. Identifying Independencies in Causal Graphs with Feedback 2 BAYESIAN NETWORKS VS . *Intelligence*, (1):43, 1988. URL http://www.cs.ucla.edu/pub/stat{_}ser/R243.pdf.

Alun Preece. Asking âĂŸWhy' in AI: Explainability of intelligent systems âĂŞ perspectives and challenges. *Intelligent Systems in Accounting, Finance and Management*, 25(2):63–72, 2018. ISSN 21600074. doi: 10.1002/isaf.1422.

PyGraphviz developer team. Pygraphviz. URL https://pygraphviz.github.io.

Pramila Rani, Changchun Liu, Nilanjan Sarkar, and Eric Vanman. An empirical study of machine learning techniques for affect recognition in human–robot interaction. *Pattern Analysis and Applications*, 9(1):58–69, 2006.

Thomas D. Schneider. Molecular Information Theory Primer. (301), 2005. URL http://users.fred.net/tds/lab/paper/primer/primer.pdf.

Jacob Schreiber. Pomegranate, a. URL https://pomegranate.readthedocs.io/en/latest/.

Jacob Schreiber. Bayesian Network Structure Learning tutorial, b. URL https://github.com/jmschrei/pomegranate/blob/master/tutorials/B_Model_Tutorial_4b_Bayesian_Network_Structure_Learning.ipynb.

Jacob Schreiber. Pomegranate: fast and flexible probabilistic modeling in python. 18:1–6, 2017. URL http://arxiv.org/abs/1711.00137.

Jacob Schreiber and William Noble. Finding the optimal bayesian network given a constraint graph. 2017. doi: 10.7287/peerj-cs.122v0.2/reviews/2. URL https://peerj.com/articles/cs-122/.

C. E. Shannon, W. Weaver, and R. E. Blahut. The mathematical theory of communication (TODO check jahr 48/49). *Urbana: University of Illinois press*, 117(April 1928):379–423, 1949. ISSN 07246811. doi: 10.2307/3611062. URL http://math.harvard.edu/{~}ctm/home/text/others/shannon/entropy/entropy.pdf.

Solomon Eyal Shimony. Finding MAPs for belief networks is NP-hard. *Artificial Intelligence*, 68 (2):399–410, 1994. ISSN 00043702. doi: 10.1016/0004-3702(94)90072-8. URL http://www.cs.columbia.edu/{~}adrian/candp/Shi94-FindingMAPbeliefNetworksNPhard.pdf.

Chandler Stolp, Shirley Dowdy, and Stanley Wearden. *Statistics for Research*, volume 3, page 637. 2006. ISBN 047126735X. doi: 10.2307/3324586.

Simone Stumpf, Vidya Rajaram, Lida Li, Weng-Keen Wong, Margaret Burnett, Thomas Dietterich, Erin Sullivan, and Jonathan Herlocker. Interacting meaningfully with machine learning systems: Three experiments. *International Journal of Human-Computer Studies*, 67(8): 639–662, 2009.

Sjoerd T Timmer, John-Jules Ch Meyer, Henry Prakken, Silja Renooij, and Bart Verheij. Explaining bayesian networks using argumentation. In *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 83–92. Springer, 2015.