POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Textual eXplanations for intuitive Machine Learning

Laurea Magistrale in Computer Science and Engineering - Ingegneria Informatica

Author: Vittorio Torri

Advisor: Prof. Mark James Carman

Academic year: 2020-2021

1. Introduction

In the last decade machine and deep learning have gained enormous popularity and they are now used in many software applications. Their widespread adoption is leading to a growing demand for explanations regarding the reasons behind the outputs of these systems, which are often black boxes. Users need to trust these systems to use them, especially in certain domains such as medicine, and even regulations are starting to introduce this requirement. Research in eXplainable AI (xAI) is trying to meet this demand by developing techniques that can bring some light inside these black boxes. Despite this, most of the currently used xAI methods can help model developers to improve their systems, but are difficult to use for end users which want to gain a clear understanding of a model's behavior. The main contribution of this work is the development of a model which can generate textual explanations for a machine learning classifier. We also develop a prototype of a question-answering (Q&A) system which could be further expanded, arriving to a fully conversational explainer. We integrate these models into a web interface, where we show also other elements which can help the users to understand the behaviour of a classifier, and we use this interface to perform a user study to evaluate our models and the interface itself.
In Section 2 we briefly present the related work, while in Section 3 we present the datasets we have used and in Section 4 we discuss the evaluation methods. In Section 5 we present our interface, in Sections 6 and 7 we discuss the explanations model and the Q&A model, with their results. Finally, in Section 8 we summarize the results and the work which remains open.

2. Related work

In the last years xAI is becoming a trending topic in research, there are many papers that have been published on this topic and some techniques have gained particular popularity. The most popular techniques are feature attribution methods, which aim to determine what is the effect of the various input features on the output, for a specific sample. Among them the most famous are LIME and SHAP [1]. LIME uses a model surrogate, i.e. it builds a linear model which locally approximates the model to be explained around the point, exploiting the fact that a linear model can be easily explained using its coefficients. SHAP instead is based on the concept of *Shapley values* from cooperative game theory, which is a measure of the contribution of a player to its coalition. Lundberg and Lee proved that SHAP is the only method of this

category that can satisfy important theoretical guarantees, such as the consistency property, i.e. $f(x) = \phi_0 + \sum_{i=1}^{M} x_i \phi_i$, where ϕ_i are the Shapley values.
Another type of xAI methods are those based on examples and in particular the ones based on the concept of counterfactual [2]. In the classification context, a counterfactual is a sample which is similar to the one to be explained, but with enough differences to alter the prediction. Many proposals have been done for the computation of counterfactuals, considering even slightly different definitions of counterfactual.
There are few works related to textual explanations. One of them is [3], where Hendricks et al. use an LSTM-based model to generate textual explanations for image classification. The main limitation is that they base their work on a dataset of textual descriptions of the images and consequently their explanations do not seem particularly correlated with the reasons behind the classifier output, but more on the discriminative elements of the various classes.
There are a couple of prototypes of chatbots for explainable AI [1], but they are both very limited in the interactions and in the type of answers they can offer to the users.
Many works about the needs of the users in term of xAI are present in the literature, we can mention [4] and [5] among many. The main elements which can be derived from them are the importance of simple and intuitive explanations, mainly local, the relevance of counterfactuals and a demand for interactive explanations, possibly adapted on the type of user which is asking them.

3. Datasets

We focus on the generation of explanations for classification tasks, in particular in the medical domain. The first dataset we consider is a cardiovascular disease dataset available on Kaggle[2]. We build our system on this dataset and then we consider a second classification dataset, the Pima diabetes dataset[3], to see if and how our model is able to generalize to different datasets.

[1] https://github.com/ModelOriented/xaibot
[2] https://www.kaggle.com/sulianova/cardiovascular-disease-dataset
[3] https://www.kaggle.com/uciml/pima-indians-diabetes-database

Another dataset that we use is the MedDialog-EN dataset[4], a large dataset of medical dialogues. We want to use a generative language model, based on GPT-2, to produce the explanations, due to the great capabilities of this model in generating text which seems to be written by humans. This dataset is useful to give a first knowledge of the medical domain to our language model, so we use it for pre-training.

4. Evaluation methods

To evaluate our results we consider both automatic metrics and a user study. Among the many automatic metrics used in NLP, we consider the following three:

1. BLEU, in particular the cumulative BLEU-4 for explanations and the cumulative BLEU-1 for Q&A
2. METEOR, whose ability to understand synonyms is particularly useful
3. BLEURT, a different type of metric which uses a transformer trained to predict human judgment over a (candidate, reference) pair

The main problem of automatic metrics is that they need a reference sentence and, in absence of a classification dataset with textual explanations, for us the reference can be only the explanation that we automatically generate, but our model may express the same concept in a different, possibly better, way, achieving a low score. METEOR and BLEURT have been chosen since they can partially overcome this problem, but the best evaluation is the one which comes from the users. Due to the difficulties in performing a user study, we leave it at the end of our development process.

5. Interface

We build an interface for our user study that can also be useful in general to show a classifier's output to the users, allowing them to better understand the reasons behind it.
A screenshot is reported in Figure 1, with some of the following main elements:

1. The data of the **current sample**, together with the classifier output
2. **Distribution plots**, showing the distribution of feature values in the dataset

[4] https://github.com/UCSD-AI4H/Medical-Dialogue-System

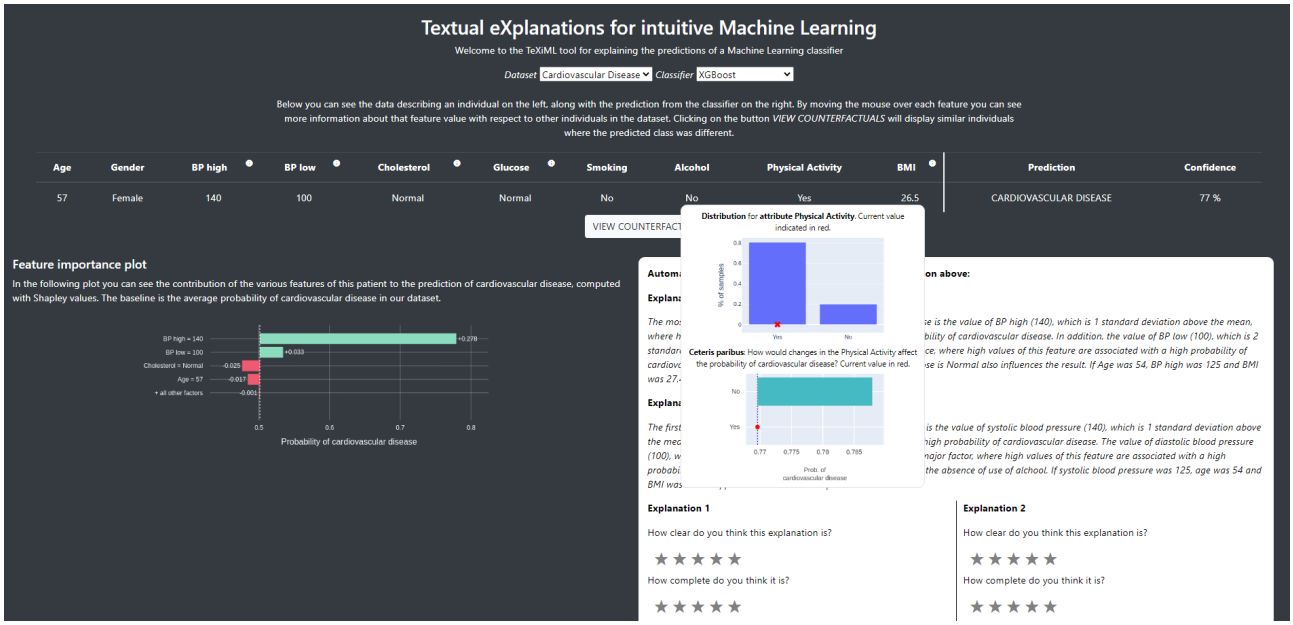

Figure 1: Screenshot of the TeXiML web interface used for the user study

3. **Ceteris paribus plots** showing what happens if the value of a feature changes, keeping fixed all the other ones
4. A **feature importance plot**, showing the effects of the various features on the result, in terms of Shapley values
5. A **counterfactuals table**, with three counterfactuals
6. Two **textual explanations**, one computed by applying the automatic rules of our grammar and the other by our language model
7. An **interaction form** for the Q&A system

Users are presented with questions regarding the completeness, correctness and clarity of the explanations, plus a choice of the preferred one, between the rule-based and the one produced by GPT-2. If they use the Q&A they can also evaluate the answer. They see samples from both datasets and at the end they are asked to answer a series of general questions about the system.

6. Textual explanations

Our goal is to generate textual explanations for the output of a black-box machine learning classifier on a given sample. This requires to define what we want to represent in these explanations and consequently on which basis they should be computed. We progressively extend our explanations:

- **Version 1**: we only express the three input features which are more relevant for the result, using as measure the SHAP method, due to its theoretical guarantees
  eg: *The prediction of disease is determined by the systolic blood pressure (140) and by the age of the patient. The BMI (29.3) also contributes to the result.*
- **Version 2**: we include additional information on mean and standard deviation of numerical features
  eg: *The prediction of disease is determined by the systolic blood pressure (140), which is one standard deviation above the mean [...]. The BMI (29.3), higher than the mean, also contributes to the result.*
- **Version 3**: includes also the description of a counterfactual
  eg: *[...] If BMI was 27 and systolic blood pressure was 130, then the prediction would have been no disease.*
- **Version 4**: includes also an explanation related to the information which comes from the ceteris paribus plot
  eg: *The prediction of disease is determined by the systolic blood pressure (140), which is one standard deviation above the mean and whose high values are associated with cardiovascular disease. [...]*

To build a training set for our language model we use a grammar with some rules which compose the explanations for our classification dataset on

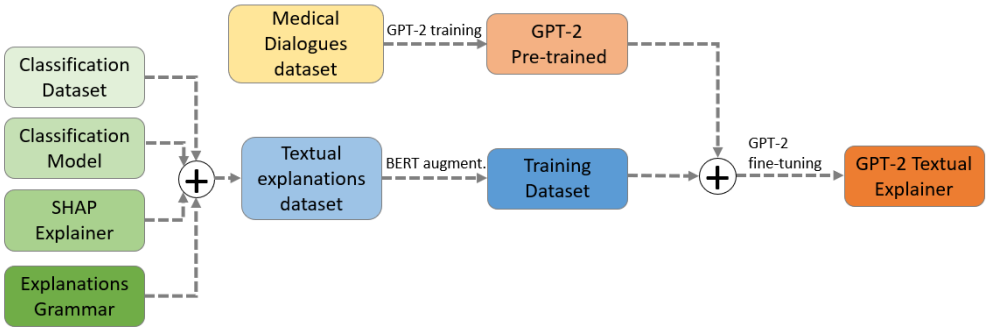

Figure 2: Schema of the training process for the textual explanations model

the base of the Shapley values, the feature values and the classifier output. In the final version (V 4*), we extend this grammar, introducing more variability in the text. We report a short portion of the grammar in Appendix, while the schema in Figure 2 represents the training process for the textual explainer.

6.1. Input encoding

The input for the explanation model needs to contain the sample to be explained, together with the information which can allow the model to understand how to produce an explanation for it. We consider different alternatives for the encoding of this data. After a comparison we select the following one:

- *input=[name=BMI, value=32, shap=0.1, mean=25, std=3.2, cp_low=low, cp_high=no]; [name=cholesterol, value=normal, shap=0.2]; [...]; prediction=no disease; cf=[name=age, value=55] [name=BMI, value=37.2]; cf_pred=cardiovascular disease*

In the first part we report the feature names and values, including the Shapley values. For numerical features we include also mean and standard deviation (from *Version 2*) and a summary of the ceteris paribus plot (from *Version 4*), with two words related to its high and low values. Then there is the classifier output (*prediction)* and the information about the counterfactual (from *Version 3*): the values which differ from the current ones and its prediction.

6.2. Augmentation

To improve the variability of the training set and avoid the language model to learn by hearth the expressions of our fixed rules, we augment it using BERT to replace randomly masked words in the explanations (with the exclusion of feature values and feature names). We consider the possibility of applying it multiple times. This leads to a decrease of the metrics, which is partially due to their difficulties in assessing the same meaning behind different sentences, and sometimes it introduces also wrong terms in the explanations, so we limit to one pass.

Ver.	BLEU-4	METEOR	BLEURT
V1	0.571	0.716	0.718
V2	0.604	0.732	0.721
V3	0.541	0.713	0.693
V4	0.590	0.756	0,670
V4*	0.527	0.713	0.640

Table 1: Results with automatic metrics of the different explanation versions on the cardiovascular disease dataset

6.3. Results

We report in Table 1 the results with the automatic metrics of the various versions of the explanations. We can observe how the inclusion of the counterfactual (V3) and the extension of the grammar (V4*) determine a reduction of the values, since they increase the complexity of the explanations. The same does not happen in the transition from version 1 to 2 and from 3 to 4, this is probably due to the fact that the information coming from the distribution and ceteris paribus plots are easier to add and they are expressed in a way more similar to the references. In Table 2 we report the results on the diabetes dataset, comparing our model trained on the

Model	BLEU-4	METEOR	BLEURT
TL	0.239	0.413	0.393
FT	0.495	0.768	0.566

Table 2: Results on diabetes dataset (V4*) with original model (TL) and fine-tuned model (FT)

Model	Dataset	Clear	Compl.	Corr.
Grammar	Cardio	3.98	**4.20**	**4.10**
GPT-2	Cardio	**4.10**	4.01	3.73
Grammar	Diabetes	**4.07**	**4.12**	**4.09**
GPT-2-TL	Diabetes	3.32	2.37	1.97
GPT-2-FT	Diabetes	3.56	3.52	3.25

Table 3: User study average evaluation of explanations properties (scores in 1-5)

cardio dataset (TL) and the same model fine-tuned on explanations for the diabetes dataset (FT). There is a significant drop of the original model which can be compensated with the fine-tuning.

In Table 3 we report the average results of our user study on the explanations. On the cardio dataset the difference is small, while on the diabetes dataset there is a clear preference for the grammar-based, even if the fine-tuned model is not far. It is interesting to observe the answers to the questions *Which explanation do you prefer?* (Table 4) and *Which type of explanation did you find more natural, on average?* (Figure 3). The users seem to appreciate more the GPT2-based explanations on the cardiovascular dataset, despite the clarity/completeness/-correctness scores do not give this clear indication.

7. Question Answering (Q&A)

Considering the demand for interactive explanations expressed in the literature, we want to offer

Model	Dataset	Pref.
Grammar	Cardio	39%
GPT-2	Cardio	**61%**
Grammar	Diabetes	**89%**
GPT-2-TL	Diabetes	11%
Grammar	Diabetes	**65%**
GPT-2-FT	Diabetes	35%

Table 4: *Which explanation do you prefer?*

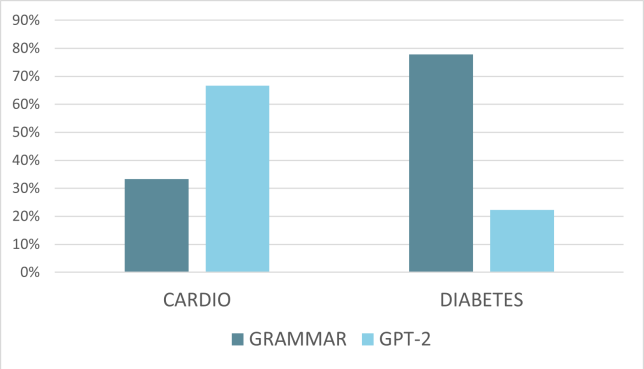

Figure 3: *What type of explanations did find more natural, on average?*

to the users the possibility to ask questions. We focus on two types of questions:

1. Feature importance questions: *What's the importance of age? What's the relevance of BMI?*
2. What-if questions: *What if age was 60? What would happen if glucose was 100?*

To develop the Q&A we use another GPT-2 based model. We build a dataset of (question, answer) pairs for the samples of our cardiovascular disease dataset. For each sample we have two pairs, one for each of the above types, created by sampling a question template from a set of predefined templates we wrote. The templates are then filled randomly with a feature and, for the second type, also with a feature value, drawn from the dataset distribution of that feature. The answer for the first type is again drawn from a set of answers, depending on the Shapley value of the feature. For the second type the procedure is slightly different: we do not expect the language model to be able to know the result of the classifier when a feature is changed, but we ask it to recognize the feature to be changed and the new value. We train it to produce an output of the type $<WHAT\_IF>feature{=}value$. Then from this output we can call the classifier on the modified sample and return a textual description of the result. We apply BERT augmentation for questions and answers of the first type and for questions of the second type.

7.1. Results

We report in Table 5 the results measured with automatic metrics. In Table 6 there are the average scores given by the users to the answers, divided by question type and dataset (for both

Dataset/ Model	BLEU-1	METEOR	BLEURT
Cardio	0.370	0.344	0.734
Diab/TL	0.308	0.269	0.441
Diab/FT	0.390	0.387	0.574

Table 5: Q&A results with automatic metrics

Dataset/Questions	Clear	Compl.	Corr.
Cardio/F	3.63	3.13	3.13
Cardio/W	3.38	3.31	3.13
Diabetes/F	4.75	3.88	1.88
Diabetes/W	4.64	4.73	4.45

Table 6: User study average evaluation of answers properties (scores in 1-5), W=What-if questions, F=feature importance questions

datasets we've used the same original model in the user study).
We can observe that the results are discrete for the cardio dataset, while for the diabetes dataset they are even better, except for the correctness of feature importance questions, where the model seems pretty weak.

8. Conclusions

We have demonstrated that it is possible to use a generative language model to produce explanations for a machine learning classifier. The performances on the cardiovascular disease dataset are pretty good, while it suffers more when it is used on a different dataset, like the Pima diabetes dataset. With fine-tuning its performances improve significantly, but they are still lower than our baseline. We believe that further work can be done to improve the generalization capabilities of the model. A possibility could be the training of the model on explanations for many different datasets, preferably all in the same domain. The richness and variability of explanations could be improved with a human-written dataset of explanations, possibly collected via crowdsourcing, which may be used for a final fine-tuning.
The Q&A system achieved discrete results on the types of questions on which it was trained and it has better generalization capabilities, but it has still a large margin of improvement, on both the correctness of the answers and on the variety of questions it can properly answer. We strongly believe that it must be a key component of an interactive explanation system.

Bibliography

[1] S M Lundberg and S Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, pages 4768–4777, 2017.

[2] S Wachter et al. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

[3] L A Hendricks et al. Generating visual explanations. In *European conference on computer vision*, pages 3–19, 2016.

[4] T Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38, 2019.

[5] Q V Liao et al. Questioning the ai: informing design practices for explainable ai user experiences. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–15, 2020.

Appendix

A short portion of the grammar rules used to automatically generate the explanations dataset:

S → *F SN T*
F → *The main reason why P has been predicted as O1 is the EF*
| *The first element which influenced the prediction of O2 is the EF*
| *The most relevant factor for the prediction of O2 is the EF*
SN → *. In addition, the EF also has a significant influence*
| *. The EF is also an important element*
| *. Moreover, the EF plays an important role*
T → *and also the EF is relevant .*
| *, while the third factor is the EF.*
| *. Finally, the EF also influences the result .*

POLITECNICO DI MILANO
Master of Science in Computer Science and Engineering
Dipartimento di Elettronica, Informazione e Bioingegneria

Textual eXplanations for intuitive Machine Learning

Supervisor: Prof. Mark James Carman

Master Thesis of:
Vittorio Torri, matr. 945208

Academic Year 2020-2021

Ringraziamenti

Innanzitutto un sentito ringraziamento va al mio relatore, il professor Carman, che mi ha guidato e supportato in questi mesi di lavoro. All'inizio non è stato facile definire con precisione gli obiettivi e la strada da seguire per questo progetto, ma il suo entuasiasmo e la sua esperienza mi hanno permesso di portarlo a termine con successo.

Un ringraziamento va anche al professor Boracchi, che ci ha fornito il suo parere e i suoi consigli.

Grazie anche a tutti i docenti che in questi anni hanno saputo trasmettere la loro passione e la dedizione per il loro lavoro, diventando fonte d'ispirazione.

Un grande grazie va a mia mamma Anna Maria e a mia sorella Rosanna, che mi hanno sostenuto durante tutti questi cinque anni, ma più in generale a tutta la mia famiglia, che mi ha sempre supportato.

Un grazie a tutti gli amici storici, in particolare Davide, Matteo, Filippo, Daniel, Riccardo e Federico, con cui abbiamo condiviso tante esperienze, ma anche a tutti coloro che ho conosciuto in questi anni al Politecnico, in particolare Nicola, che mi ha accompagnato anche in questi mesi di tesi.

Un ringraziamento finale a tutti coloro che hanno partecipato allo user study, fondamentale per la valutazione dei risultati di questa tesi, in particolare a Matteo, Nicola e Gabriele per le loro preziose osservazioni.

Abstract

In the last decade machine and deep learning have gained enormous popularity and are now used in many software applications. Users need to trust these systems to use them, especially in certain domains such as medicine, and since most of them are black boxes there is a growing demand for explanations regarding the reasons behind their outputs. Research in eXplainable AI (xAI) is trying to meet this demand by developing techniques that can bring some light inside these black boxes. Despite this, the currently most used xAI methods are usually more helpful for model developers and experts than for end users. The main contribution of this work is the development of a system which can produce textual explanations for a machine learning classifier. We automatically generate a dataset of textual explanations for a classification problem and we use it to train the well-known GPT-2 model to produce text to explain a classification model's predictions. Considering the large demand for interactive explanations, we introduce also a question answering system (Q&A), able to answer questions about (the reasons for) the model's predictions. We build an interface, which can be easily used for many classification tasks, where we present the textual explanations and the Q&A system, together with other elements that can help the users to understand and trust a classifier. We finally use this interface to collect evaluations of our system through a user study. The results highlight the effectiveness of our explanations on the dataset we have used for the development of the system and the limitations of a direct porting on a different dataset, which can be largely overcome with a fine-tuning process.

Keywords: explainable AI, textual explanations, interactive explanations, xAI interface

Sommario

Nell'ultimo decennio machine e deep learning hanno guadagnato un'enorme popolarità e sono oggi usati in molti software. Gli utenti hanno bisogno di fidarsi di questi sistemi per usarli, specialmente in settori come quello medico, ed essendo principalmente "scatole nere" vi è una crescente domanda di spiegazione delle ragioni dietro i loro output. Le ricerche sull'explainable AI (xAI) stanno provando a rispondere a questa domanda sviluppando tecniche che possono portare luce dentro queste "scatole nere". Ciò nonostante, i metodi xAI attualmente più usati sono spesso più utili per gli sviluppatori di modelli che per gli utenti finali. Il contributo principale di questa tesi è lo sviluppo di un sistema in grado di produrre spiegazioni testuali per un classificatore machine learning. Generiamo automaticamente un dataset di spiegazioni testuali per un problema di classificazione e lo usiamo per il training del noto modello GPT-2, rendendolo in grado di produrre spiegazioni testuali relative all'output di un classificatore. Considerando la grande richiesta di spiegazioni interattive, introduciamo anche un sistema di question answering (Q&A), in grado di rispondere a domande sugli output di un classificatore. Costruiamo un'interfaccia, che può essere facilmente utilizzata per vari problemi di classificazione, dove presentiamo le spiegazioni testuali e il sistema di Q&A, insieme ad altri elementi che possono aiutare gli utenti a comprendere l'output di un classificatore. Infine utilizziamo questa interfaccia per raccogliere valutazioni sul nostro sistema. I risultati evidenziano l'efficacia delle nostre spiegazioni sul dataset che abbiamo utilizzato per lo sviluppo del sistema e le limitazioni di un porting diretto su un dataset diverso, che possono essere in buona parte superate con un processo di fine-tuning.

Parole chiave: explainable AI, spiegazioni testuali, spiegazioni interattive, interfaccia per xAI

Contents

Chapter 1

Introduction

1.1 Research area

In the last decade artificial intelligence (AI) entered for the first time in our everyday life: it is enough to think about voice-controlled devices, recommendation systems which suggest us new contents to watch, image-recognition systems, self-driving cars and many other services with which we interact constantly (see Figure 1.1 for some examples).

Despite its recent spread, the concept of *Artificial Intelligence* has been present since the birth of Computer Science, considering that its origin can be tracked back to the 1956 Darmouth Conference:

> *"The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it" [1]*

The difficulties in overcoming certain mathematical problems and the initial absence of the great results that were expected lead to the so-called *AI Winters* (1973-1982, 1987-1997), but from the end of 1990s a new summer started, where new techniques, a larger amount of available data and an increased computational power lead to the advent of *Machine Learning (ML)* [2].

In a few years, starting with 2012 Krizhevsky's *AlexNet* [3], *Deep Learning (DL)* emerged with astonishing results that produced a new faith in the

possibilities of AI, including the always dreamt idea of a *general artificial intelligence*, capable of reproducing the human mind, even if we are still far from this and there is a lack of consensus among experts about if and when this will be achieved [4].

The solutions of the first AI era were mainly based on logic inference (eg: *All men are mortal AND Socrates is a man* $\implies$ *Socrates is mortal*) and on rules (eg: *If obstacle in front, then turn right*) and because of this it was relatively easy to explain the behaviour of those systems: it was enough to list all the inferences or all the rules used to reach the final result. They could be many, but they were interpretable by humans.

This is not the case for most of the Machine Learning techniques. Even if there are some simple techniques, like decision trees or linear regression, whose models can be directly interpreted by humans, they are too weak for many tasks. Most ML models are black-boxes and it is hard to explain their internal behaviour to a human. The situation becomes even worse with Deep Learning: the simplest deep neural networks can already have tens of millions of parameters, while the more complex ones can reach billions of parameters (eg: AlexNet has 60 million parameters [3], GPT-2 has 1.5 billion parameters [6]). The need for understanding the reasons behind the behaviours and the outputs produced by these complex models lead to an area of research which goes under the name of *eXplainable AI* (xAI). Various techniques have been developed in the last years, but the problem is far from being fully solved.

The goal of this work is to explore the possibility of providing explanations in natural language, easily understandable for non-technical users, using generative models like GPT-2. Moreover, it aims to develop an interface which can be used as a support tool for users of different classification tasks, where the textual explanation is a key component but not the only one.

We consider the state of the art in xAI and the limitations of the current solutions, together with the needs expressed by the users of AI-based systems. We focus on the classification problem, one of the most common in machine and deep learning, and we develop a model able to provide natural language explanations. This model is then embedded in a web interface which can be easily used for different classification problems, showing to the users the

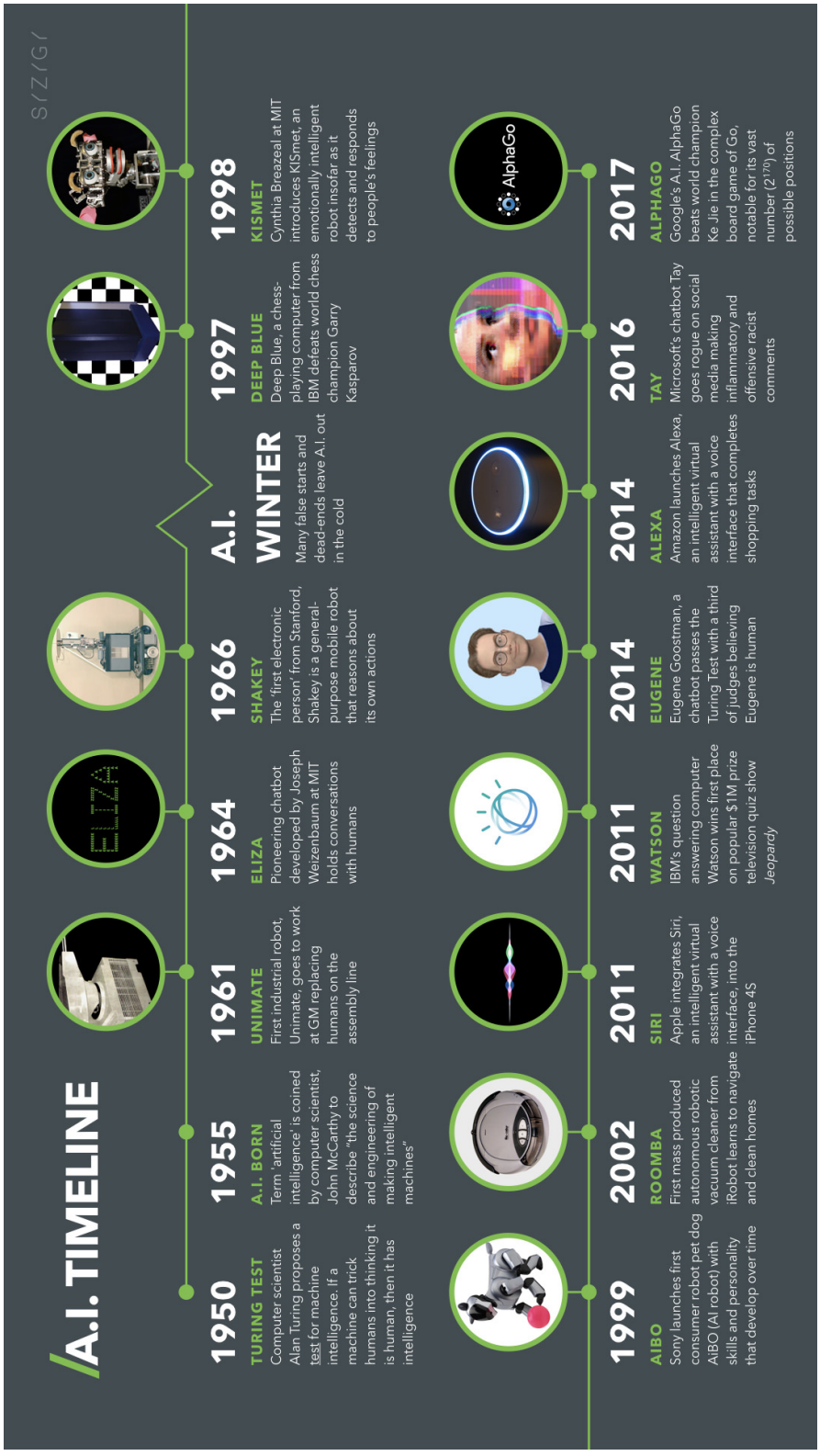

Figure 1.1: A timeline of AI with some famous products. [5]

data and the textual explanation, together with various elements which can help in understanding the data used and the results produced by a machine or deep learning model. We also investigate the possibility of an interaction of the users with the explanation system.

1.2 Work description

The main type of models on which this work is focused are classification models for datasets with numerical and/or categorical features. They are one of the most common problems in machine and deep learning and there are many examples of domains where they are used, like medicine or finance.

We analyze in detail the main techniques which have been proposed so far in the literature to explain these types of models, in particular the black-box ones, i.e. the ones which can be applied to any classification model. We consider what has been done to realize systems which can help non-technical users of these models to understand and trust (or mistrust when it is the case) them.

Among the different techniques, we focus in particular on the use of the concept of *Shapley values*, which originally comes from game theory, as proposed by Lundberg and Lee [7], and on the idea of counterfactual explanations. These tools will be the base for generating textual explanations in natural language, which is the main innovative contribution of this work. We initially consider a specific dataset in the medical domain, for which we generate in an automatic way the textual explanations. This dataset, with the automatically-generated explanations, will be the training set for a language model whose purpose is to generate textual explanations in the most natural way possible. Later we apply the same methodology on another dataset, comparing the results.

Finally, we embed these explanations in a web interface which is also used to collect feedbacks from the users about the system and the explanations it proposes.

Future research has to be made to extend the interaction with the users. We propose a first prototype of a question-answering system, currently working only on very specific types of questions and that could be extended, until the ideal point of having a fully conversational explainer.

1.3 Document structure

The document is structured as follows:

- In Chapter 2 we present some background knowledge about the topics of this thesis and the state of the art, showing the main techniques for explainable AI and the researches which have been conducted about the needs of the users in terms of explanations for machine learning models.
- In Chapter 3 we present the research questions we want to address.
- In Chapter 4 we present the datasets we used in our work.
- In Chapter 5 we describe our approach, with the models and the techniques we used.
- In Chapter 6 we show the results of our experiments, including a subjective evaluation obtained through a user study.
- In Chapter 7 we summarize the goals of this work, we discuss the answers that we have found for the research questions presented in Chapter 3 and the work which remains open.

.

Chapter 2

Background & Related work

2.1 Machine & Deep Learning

We provide a brief introduction to the very large fields of Machine and Deep Learning, whose basic knowledge is necessary to understand this work.

2.1.1 Machine Learning

Machine Learning (ML) can be considered a subset of the broader field of Artificial Intelligence (AI), particularly focused on building algorithms which can improve their performances with the experience and in particular which learn from data [8]. Learning from data is the key point which makes machine learning different from the so called "traditional" AI.

Traditional AI was mainly based on knowledge bases of rules and axioms and exploited logical reasoning on these knowledge bases to produce its "intelligent" behaviour. Machine Learning instead bases its "intelligence" on the data. There are three main subfields of machine learning, depending on the type of data which are used:

1. **supervised learning**: it is based on the use of a *training set*, i.e. a set of annotated examples, from which it can learn a function which can be used to predict the labels for previously unseen data. Its main tasks are *regression*, where the labels are continuous numbers (eg: predict the income of a company), and *classification*, where the labels are discrete numbers (eg: distinguish patients affected or not from a certain

disease).

2. **unsupervised learning**: it is based on a set of unlabelled data, its main tasks are *clustering*, where the data are divided into groups on the base of some kind of similarity (eg: customer segmentation), and *dimensionality reduction*, where the number of features (i.e. characteristics, properties) of the data are reduced to keep only the most relevant ones.

3. **reinforcement learning**: an agent has to learn a policy which governs its behaviour in a certain environment on the base of the positive or negative rewards it receives from the environment after each action.

In this work we will deal with supervised learning and in particular with classifications tasks. There are many machine learning models for classification, we will use some of them, in particular *Logistic Regression, XGBoost and Random Forest.*

2.1.2 Deep Learning

The main limitation of Machine Learning is related to the difficulty in finding the best features which need to be given as input to the machine learning model. In many cases the properties of the data that we can directly access, i.e. their representation, are not necessarily the best direct input for a machine learning model. There are various algorithms which allow to select among the available features, but for many tasks the right features need to be derived somehow from the raw data representation in non trivial ways. An example where this is very simple to see are images: the pixels, which are the raw representation of the input, are not particularly meaningful in themselves, but instead there are portions of the image which can be meaningful, depending on the task, and it is quite hard to identify them.

Deep Learning solves this problem, providing models which are able to work with the raw representation of the data and to solve complex tasks. These models are called **Artificial Neural Networks** (ANN), or simply Neural Networks (NN). A Neural Network is a mathematical model which can be represented as a graph. There are different types of neural networks,

in this section we describe only the simplest ones, Feed Forward Neural Networks (FFNN), but the idea behind more complex models (CNN, RNN,, ..) is similar, with some extensions. Later we will briefly dicuss about RNNs and a more advanced neural network model called *Transformer*.

Neural Networks are mainly used in a supervised learning way, even if there are also unsupervised learning tasks which can be performed with neural networks. A FFNN is composed of a series of layers, in particular the first one is the *input layer* and the last one is the *output layer*, while the others are called *hidden layers*. Each of these layers has a certain number of units, called *neurons*. Each neuron receives all the outputs of the neurons of the previous layers, each of them with a certain weight. These weights are the key components of the network, they are what is learnt during the training phase. The i-th neuron in the j-th level computes its output as:

$$h_i^{(j)}(\sum_{k=0}^{K} w_{ki}^{(j-1)} \cdot h_k^{(j-1)})$$

where $h_i^{(j)}$ is a function called *activation function* of the i-th neuron of the j-th layer, K is the number of neurons of the previous layer, $h_k^{(j-1)}$ is the output of the k-th neuron of the previous layer and $w_{ki}^{(j-1)}$ is the weight connecting the k-th neuron of layer $(j-1)$ to the i-th neuron of layer j.
An example of a simple FFNN with a single hidden layer is presented in Figure 2.1. A neural network for binary classification has typically a single output, which corresponds to the probability of the "positive" class, while in case of k-classes classification it has k outputs, one for each class, indicating the probability of that specific class. For regression tasks they have an output for each variable which has to be predicted.

The popularity of deep learning has exploded in the last decade, but its origins can be traced back to the the 1950s, when Rosenblatt proposed a model called *Perceptron* [9], the first model of an artificial neuron. This model was still quite limited and it was only with the development of the backpropagation algorithm [10] that it became possible to effectively use neural networks. An important result is the *Universal Approximation Theorem*, firstly stated by Hornik et al. [11] but later extended, whose meaning is basically that for any function there is a FFNN which can approximate it with

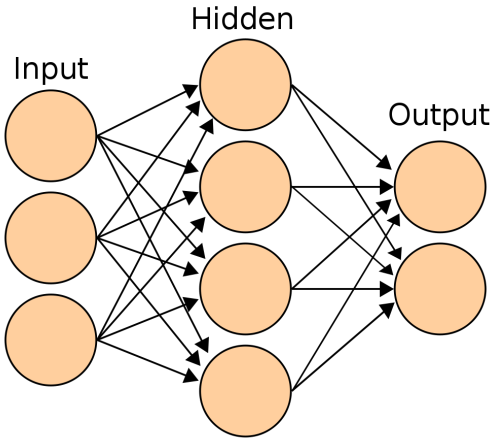

Figure 2.1: A simple FFNN [1]

any desidered degree of accuracy [12].

The factors which have allowed the recent success of deep learning are mainly two:

1. the availability of **big data**: in the last decade the enormous amount of electronic devices and services that have widely spread all over the world has allowed to collect very large quantity of data, which are fundamental for the performances of deep learning models

2. the increased **computational capacity**, in particular in term of GPUs (*Graphic Processing Units)*, which are necessary to train deep neural networks in a reasonable amount of time

2.2 Language Models

Language models are the base of *Natural Language Processing (NLP)*, their purpose is to represent the joint probability of sequences of words. Since the number of possible word sequences is enormous, traditional language models are based on the concept of *n-gram*: they model the conditional probability of a word given the $n-1$ previous words. Considering that the number of words in a language can be in the order of millions (10^6), a 10-gram would

[1] By Cburnett - Own work, CC BY-SA 3.0, `https://commons.wikimedia.org/w/index.php?curid=1496812`

have to learn $(10^6)^{10}$ probabilities, which is clearly unfeasible. Because of this n is typically restricted to 2 or at most 3, even if this implies a stronger assumption which reduces the performances.

To overcome the curse of dimensionality that is present in the simple *bag of words* representation (where each words is represented with a binary vector whose length is the length of the dictionary), neural networks have been used to develop a new type of word representations called *word embeddings*. The idea behind these representations is to train a neural network to predict the next word probability given the previous n-1 words (or the n-1 surrounding words) and then use the internal representation of the input obtained in an hidden layer as a representation for the missing word. This allows a continuous representation which takes into account the context in which the word is present and which allows similar words to have similar representations. There are many word embeddings, the original idea is from Bengio et al. [13] but the first successful implementation is Google's Word2Vec [14].

2.2.1 Recurrent Neural Networks

When dealing with certain tasks it can be useful to have the ability of keeping a memory of the previous input, i.e. a state. This is the reason behind recurrent neural networks (RNN). RNNs are neural networks where there are also *recurrent connections*: the neurons can receive in input also their output at the previous step. Figure 2.2 shows the RNN architecture.

This can be useful in tasks like machine translation, where the input are the various words of a sentence and we want to produce a translation of the input in an another language: every output word is not only related to the corresponding input word, but to the entire input sentence. In general all the tasks where there is a sequence of inputs which affect the output can obtain benefit by the concept of RNNs. Unluckily RNNs suffer of two technical problems called vanishing and exploding gradient, which limited their use until a new architecture called LSTM was proposed [15].

2.2.2 Transformer architecture

The transformer architecture [16] is an architecture which has been able to overcome the limitations of memory-based architectures like LSTM, where

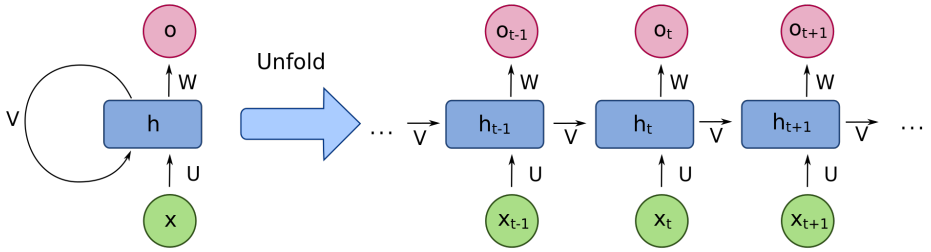

Figure 2.2: A simple RNN on the left. The same RNN unfolded over time on the right[2]

.

it was hard to have parallelization during the training due to the sequential nature of the model. Transformers-based models, like BERT [17] and GPT-2 [6], are now the state of the art for many NLP tasks.

The transformer is composed of a stack of modules called encoders and another stack of modules called decoders. Both types of modules are based on the concept of attention, which was already introduced in LSTM and allows to focus on specific portions of the input. The absence of the recurrent connections could lose the order of the elements in the input sequence, but this is avoided thanks to a positional encoding, which alters the input embedding so to encode also the position of each element in the sequence. A schema of the transformer architecture is in Figure 2.3.

2.2.3 GPT-2

GPT-2 [6] is a transformer-based language model which has demonstrated the power of unsupervised training over an enormous corpus of text. The model has been trained in a unsupervised way with 40 GB of text crawled from the web following links from Reddit posts and it has been able to achieve relevant results in many tasks, among which there are machine translation, question answering and summarization, improving the state of the art in some of them, without any fine-tuning. It is able to generate text that is often hard to distinguish from human-written. Moreover, it has been used also as a

[2]By fdeloche - Own work, CC BY-SA 4.0, `https://commons.wikimedia.org/w/index.php?curid=60109157`

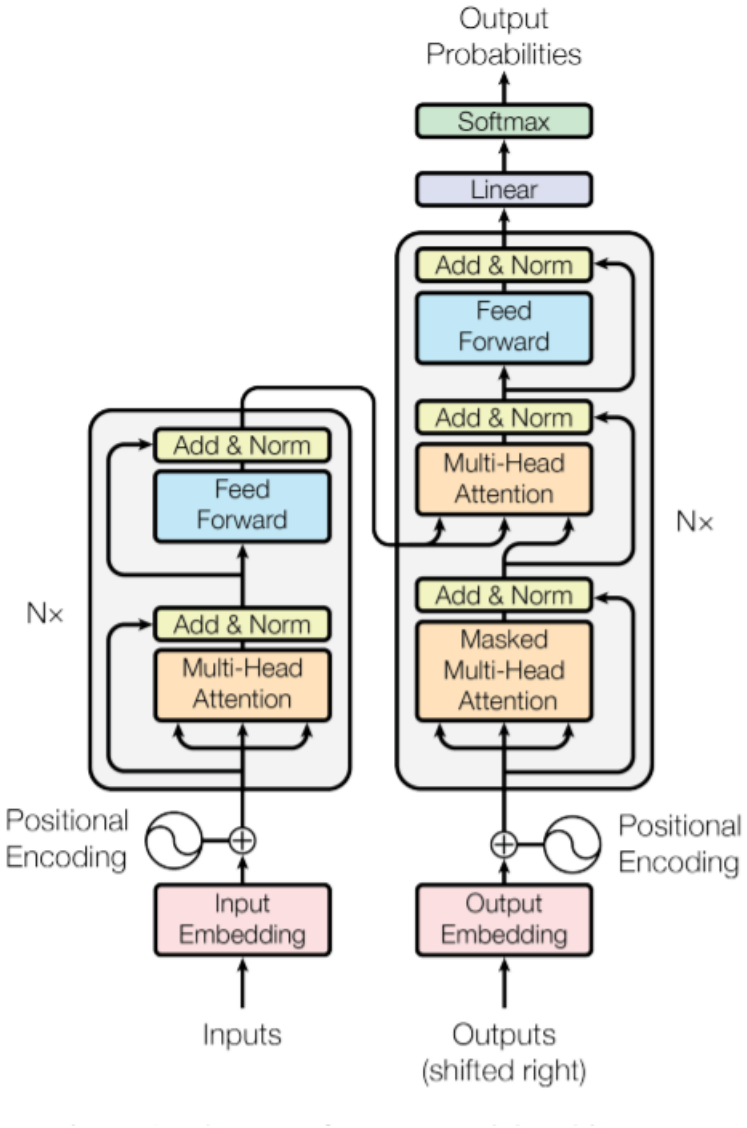

Figure 2.3: The transformer architecture (from [16])

base for more specific tasks, with fine tuning, like in [18] and [19]. Open-AI has also develop its successor, GPT-3 [20]. Its architecture is very similar to the one of GPT-2, but the size is increased, reaching 175 billions parameters (vs 1.5 billions of GPT-2). Its source code has been exclusively licensed to Microsoft and its not currently available, it can only be used via APIs, previous authorization. This is the first reason why we use GPT-2, the second one are the difficulties which arise in terms of computational requirements for fine-tuning but also for doing inference with such large models.

2.3 eXplainable AI

As we have mentioned in Chapter 1, most of machine learning models and all deep learning models are very hard to interpret and explain to a user. They are typically seen as magic black-boxes which receive an input and produce an output, without any information of what happens in the middle, and this lead to a field of research called *eXplainable AI*.

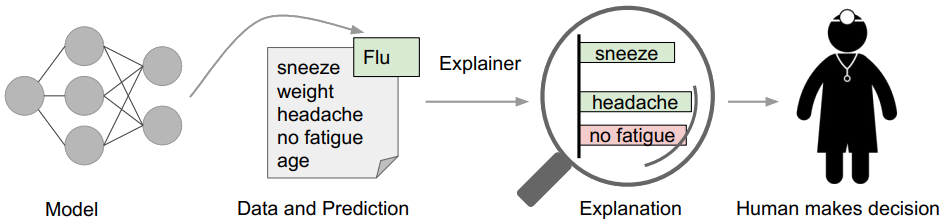

Figure 2.4: xAI pipeline (modified from [21])

2.3.1 Origin of xAI

The field of explainable AI (*xAI* in short) has mainly developed in the last decade, even if the idea of explainability was already present since the first steps of AI in the previous century.

The first AI-based systems, called *expert systems*, were mainly based on logical inference, starting from a knowledge base of axioms and rules. Thanks to this structure they were inherently explainable: it was enough to follow the reasoning procedure of the system and show it. Clearly there were different

types of questions that could be asked to get explanations and it was not always immediate to reconduct them to the reasoning of the system, but there were proposals to make this simpler, like the *EES Framework* of Neches et al. [22].

Another interesting example of explainability in a more recent expert system is the one included in the *Full Spectrum Command (FSC)* simulator, a kind of RTS videogame used to train US soldiers [23]. The behaviour of the AI-controlled player is based on a series of rules derived from the military tactics, with some simplifications. After each mission the game includes a debriefing phase were the xAI module comes into play. During the mission all the relevant events are recorded, then the user can ask questions from a predefined set and the recorded events, together with the rules in the knowledge base, allow to answer them. It is worth noting that the predefined questions have been selected so to avoid to disclose to the user some tactics followed by the AI which does not follow the US Army doctrine. The paper about this system is also the first one where the term *eXplainable AI* has been used, although it still required a few years to become a popular topic.

2.3.2 Development of xAI

The big increase of attention about xAI is related to the development of Machine and Deep Learning, whose models are mainly black-boxes and so their explanation is much more challenging. There are a few exceptions, like *decision trees*, where the structure of the tree can provide an explanation of the output, or *linear regression*, where the coefficients of the features can be used to determine their effect on the output. Apart from these, the majority of the models are not inherently explainable. In general there is a tradeoff between interpretability and accuracy, depicted in Figure 2.5 . We can think for instance to all the ensamble methods or to neural networks: they are far too complex to explain their internal behaviour to a user. Because of this, new methods have been developed by xAI researchers, to be able to provide insights about their behaviour.

There are different types of methods which have been developed in xAI. A first distinction has to be made between **global** and **local** explanations: a global explanation is a general explanation of the behaviour of the model with respect to the input it receives, while a local explanation is an explanation

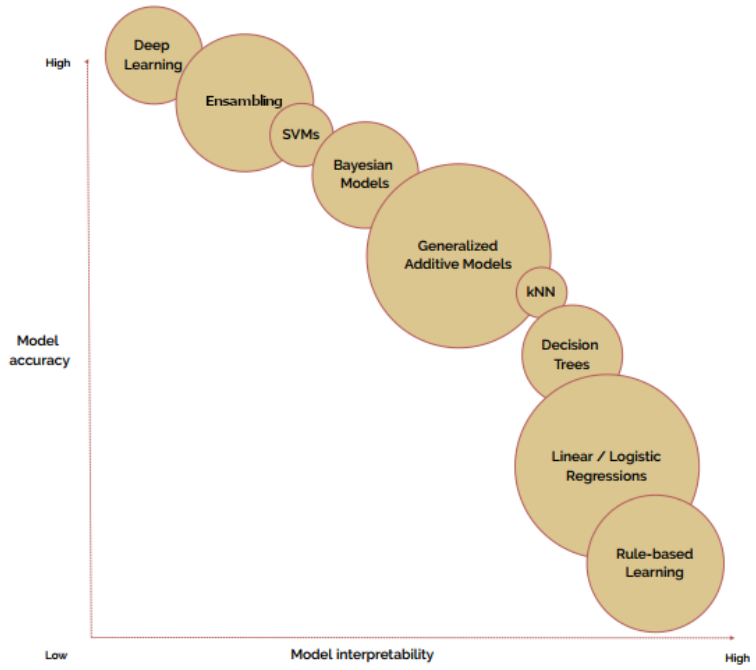

Figure 2.5: Interpretability vs Accuracy tradeoff in AI (from [2])

of the reasons behind the output for a specific input. Another distinction has to be made on the base of the type of explanation. The majority of xAI techniques are **feature attribution** methods, i.e. they assign to each input feature a weight, possibly negative, which is somehow related to its effect on the output. Other techniques instead are based on **examples**: they explain the reasons behind a prediction providing similar examples from the dataset which are classified in the same way or in the opposite way. Then there are methods which produce **plots** from which a user can derive some insights about the reasoning that is done by the model. In the next sections we present the most important techniques, some of which are the base for this work.

2.3.3 Feature attribution methods

LIME (Local Interpretable Model-agnostic Explanations)

LIME is a technique proposed by Ribeiro et al. in [21] to explain every type of ML model building a **local surrogate** of the model. For a given data

point LIME builds a local linear model around it and then this local linear model is used to produce the explanation since, as we previously mentioned, it is a simple model which is inherently explainable. LIME is based on a binary representation of data points and from this representation it samples other data points by perturbing the original one. It labels all these points with the original model output for them and then it fits a local linear model by optimizing the following type of expression, where f is the original model, g is its local approximation, L is a loss function, π is a proximity measure between the points and Ω is a measure of the complexity of the model, like the number of non zero-weights:

$$arg \min_{g \in G} L(f, g, \pi) + \Omega(g)$$

In particular they use a K-Lasso, where K is a fixed number of features they want to obtain. This procedure selects the K features with Lasso and then it fits a linear model with these K features, using a weighted least-squares, where the weights are given by π, which is an exponential kernel defined on a distance function which depends from the domain. Figure 2.6 depicts the local approximation made by LIME. LIME can be easily applied also to text and images. This method is a local explanation technique, but its authors propose also a procedure to derive a global explanation. They consider the local explanation on each point of the dataset and they use a particular pick procedure to select the B best features which explain more instances.

Gradient-based methods and DeepLIFT

DeepLIFT (*Deep Learning Important Features*) is a technique proposed by Shrikumar et al. in [24], which is an evolution of a series of techniques which are all based on the concept of gradient and that can be applied to neural networks [25] [26] [27].

The idea of this class of methods is to consider the gradient of the output of a neural network with respect to the various input features and consequently determine their importance for the prediction of the current sample. Despite the simplicity of this idea, there are different problems which arise in its implementation. In particular the backpropagation through a RELU can

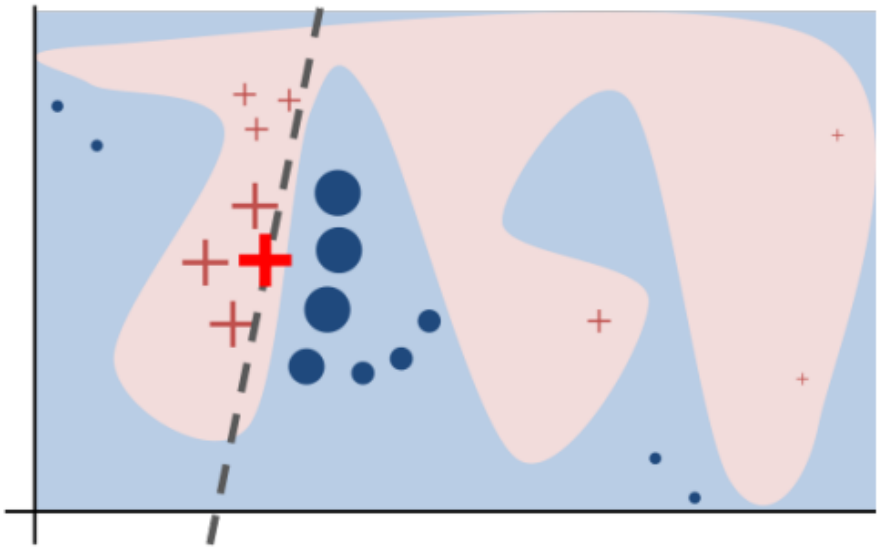

Figure 2.6: An example of the application of LIME. The big red cross is the point to be explained, the smaller points are its perturbations used to fit the local linear model which is represented by the dashed line. In the background there is the space partitioning of the model which is being explained. (from [21])

be managed in different ways: zeroing it when the input to the RELU in the forward pass was negative, zeroing it when the input in the backward pass is negative, or both. These approaches are consequently unable to highlight inputs which have a negative contribution to the output. Moreover, they suffer of problems when there is a discontinuity in the gradient and when there is a situation of saturation of a signal.

The approach of DeepLIFT is slightly different: it defines for each target neuron a reference output t and then it computes the difference Δ_t between the current output and the reference output. The weights of the inputs are such that their sum equals Δ_t. The main advantages of this method is that the weights assigned to the inputs remain continuous and that they can have non-zero weights even when the gradient is zero. The reference output is defined as the output obtained from the reference input. Consequently the definition of reference input is the critical point.

In the paper the authors present some reference inputs for different datasets, which are derived empirically comparing different possibilities. The absence

of a way to automatically define this reference input is one of the main limitation of this method. Moreover, it applies only to neural networks.

There are also gradient methods which can be applied to any classifier, like [28] which uses Parzen windows to have an estimate of a local probability function which can be derived, but it has still some limitations.

SHAP

SHAP (SHapley Additive Explanation) is a model-agnostic explanation method proposed by Lundberg and Lee in [7]. It is an additive explanation method, i.e. it satisfies the following property:

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i$$

where the z'_i are the simplified feature values (binary), ϕ_i is the weight, i.e. the effect, of the simplified feature i, and g is an approximation of the model f to be explained. The authors identify three reasonable properties which can be asked to an additive feature attribution method:

1. **Local Accuracy**: $f(x) = \phi_0 + \sum_{i=1}^{M} x_i \phi_i$, i.e. g=f
2. **Missingness**: $x_i = 0 \implies \phi_i = 0$
3. **Consistency**: let $f_x(z') = f(h_x(z'))$, where $h_x(z')$ is the mapping from the simplified binary version z' to the original version x, and let $z' \setminus i$ the instance equal to z' but with $z'_i = 0$, then for any two models f and f':

$$f'_x(z') - f'_x(z' \setminus i) \geq f_x(z') - f_x(z' \setminus i) \quad \forall z' \implies \phi_i(f', x) \geq \phi_i(f, x)$$

And there is a theorem which states that there is a unique solution which satisfies all these three properties, which is the following:

$$\phi_i(f, x) = \sum_{z' \in x'} \frac{z'!(M - |z'| - 1)!}{M!} [f_x(z') - f_x(z' \setminus i)]$$

where the sum is over all the possible z' which are subset of the non-zero entries of x', $|z'|$ denotes the number of non-zero entries in z' and M is the

number of features. In practice this considers the effect which is due to the addition of a certain feature, averaged over all the possible feature orderings. In fact, the contribution of a feature to the model output depends on the order in which we consider the features.

This theorem is a central result in cooperative game theory and these values assigned to the features (which in cooperative game theory would be the players of a coalition) go under the name of ***Shapley values*** [29].

The main problem with this method is the computation of the Shapley values, which is quite expansive applying the above formula. There are different ways in which this computation can be approximated. In [30] Štrumbelj and Kononenko propose the method described by Algorithm 1.

Algorithm 1 Algorithm for approximated Shapley values computation, from [30]

$\phi_i(x) = 0$
for 1 to m **do**
  select at random a permutation O of the features
  select at random $w \in X$
  build b_1 as a copy of x for the features which precede i in O and also for i, while instead it is a copy of w for the other features
  build b_2 as a copy of x for the features which precede i in O, while instead it is a copy of w for the other features
  $\phi_i(x) = \phi_i(x) + f(b_1) - f(b_2)$
end for
$\phi_i(x) = \frac{\phi_i(x)}{m}$

Another possibility proposed by Lundberg and Lee is the so-called Kernel SHAP. The idea behind this algorithm starts from the observation that the equation which is minimized by LIME is the one of an additive model. The choice of L, π and Ω made by LIME does not lead to the Shapley values, but since the above mentioned theorem guarantees that there exists a unique additive method which satisfies the mentioned properties and that method is SHAP, there must be a choice of L, π and Ω which leads to the Shapley values. It turns out that L can be choosen to be a squared loss and Ω can be set to zero. Using the kernel below it becomes possible to compute the

Shapley values by solving a weighted linear regression problem:

$$\pi_{x'}(z') = \frac{M-1}{\binom{M}{|z'|}|z'|(M-|z'|)}$$

The approximation which is still present is related to the fact that most models do not allow to have missing values in input, so there is the following assumption $f(z_s) \approx E_{\overline{z_s}}[f(z)] \approx f(z_s, E[\overline{z_s}])$, where $\overline{z_s}$ are the missing features, which implies the feature independence and the model linearity.

There are also some model-specific approximations and it can be demonstrated that DeepLIFT can be seen as a way to approximate Shapley values in a neural network.

A comparison between LIME, SHAP and DeepLIFT made in [7] shows that SHAP is the one which provides the highest consistency with human intuition.

2.3.4 Counterfactuals

A completely different class of explanations are the example-based explanations, where one (or more) samples are produced to justify the output of the current sample. In particular an important class of methods of this type for classification problems are the so called **counterfactual explanations**. A counterfactual is a data point which is classified in a different way with respect to the current sample, but it is not far from it. In particular, it should be as near as possible. The definition of counterfactual is not exactly unique, often these points must belong to the original dataset, but sometimes this request is relaxed.

An introduction to counterfactuals is presented by Wachter et al. in [31]. They propose a comparison between Euclidean and Manhattan distance, in the unnormalized, normalized with standard deviation and normalized with MAD (Median Absolute Deviation) versions. They suggest the use of the Manhattan distance (L1 norm), since it induces sparse solutions, where only a few features change their value, with respect to the Euclidean distance. The normalization is important to avoid the effect of features with larger values and the use of MAD instead of standard deviation makes the procedure less sensible to outliers. They consider counterfactuals as one of the best way to provide explanations which can work with any model and can be understood

by every user. They also discuss about the GDPR "*right to explanation*", mentioned in the European law with reference to all the automatic decision-making systems. Even if the text of the law is not very clear and precise and not legally binding about this topic, they consider counterfactuals as a possible way to satisfy its requests.

Another proposal about counterfactuals has been done by Rathi [32]. He proposes the use of SHAP (see 2.3.3) as a base for calculating counterfactuals. Given the class to which the counterfactual should belong, this method computes the Shapley values of the current sample with respect to that class. Then it considers the features which have negative Shapley values: these are the ones which should be altered to obtain the desidered counterfactual class. In particular, it considers a neighbourhood of the current sample and it tries to modify subset of these features values, in the current data point, with the values of the points in its neighbourhood, until it finds a mutation which is predicted with the desired class. The author considers as a positive element the fact this method is able to generate counterfactuals which are not present in the original dataset, but this is a risk: it may generate points which are not realistic. The mutation approach should limit this possibility, but it still may happen.

Poyiadzi et al. present an algorithm for counterfactual generation called FACE (*Feasible and Actionable Counterfactual Explanations*) [33]. They take into account constraints which can be imposed on the variation of certain features and they build a weighted graph of the dataset where the weights depends on the constraints, on the distance and also on the density of the path. They consider better counterfactuals the ones which can be reached with a path of high density.

Depending on the application domain these considerations may be worthwhile or not. In particular, it depends on the purpose of the counterfactual explanation: if we want to suggest to a user something s/he could change to alter the prediction, then it makes sense to impose constraints on what can be changed, but if instead we want just to give an (indirect) explanation of the reasons behind the current output, this is not necessary.

Another technique to generate counterfactuals is proposed by Looveren and Klaise [34]. They propose to find counterfactuals by minimizing a func-

tion of the following form:

$$L = c \cdot L_{pred} + \beta \cdot L_1 + L_2 + L_{AE} + L_{PROTO}$$

where

- $c \cdot L_{pred}$ is related to the difference between the prediction of the original class and the counterfactual class for the counterfactual point
- $\beta \cdot L_1 + L_2$ is related to the distance between the original point and the counterfactual
- L_{AE} is related to an autoencoder reconstruction error, to penalize counterfactuals out of the original distribution
- L_{PROTO} is related to the distance between the counterfactual and a prototype instance of the counterfactual class

They also propose to remove the L_{pred} term, speeding up the computation, since the L_{PROTO} can already lead the counterfactual to belong to the appropriate class.

The main limitation of this algorithm is the need for an autoencoder trained on the dataset to compute the reconstruction error, even if the authors propose also an alternative method based on kd-trees.

They also propose two interpretability measures for counterfactuals, based on reconstruction errors of different autoencoders, but they are not used for a direct comparison with other counterfactual generation methods.

2.3.5 Users' needs

A key question to ask when dealing with explainable AI is: *what are users' needs?* This question is fundamental and it implies another question: *who are xAI's users?* The answer to the latter is not unique, it depends from the specific application or use case. It can be that explanations are directly used by model developers and testers to help them in understanding the behaviour of their models, or it can be that explanations are used by end users of an application, or both. Once the answer to this question has been defined, then it becomes possible to find the answer to the first question and consequently determine the type of explanations and the way in which they are provided.

It is clear that certain types of explanations, like feature attribution methods, can be very useful for model developers which can easily understand them, but it may be that they are not enough clear for a non-technical user. One of the current limitations of xAI is that many of its methods have been developed having in mind only machine learning developers.

Adadi and Berrada [35] distinguish four different motivations which can drive explanations:

1. **justify**: this is mainly referred to end-users of a system, which need to understand the reasons behind the output

2. **control**: this refers to model debugging by developers

3. **development**: this refers to the improvement of the model by developers

4. **discover**: this refers to the the possibility of discovering interesting rules and patterns which are learnt by the model and that are previously unknown

Ribera and Lapedriza [36] support a user-centric approach to xAI, with different types of explanations and evaluations, depending on the users to which the explanations are produced for. Their considerations are summarized in Figure 2.7. We can note the presence of counterfactuals and plain language for lay users.

Liao et al. [37] try to identify user needs in term of explainability by asking not directly to AI end-users, but to UX and design practitioners. The idea is that there is currently a gap between algorithmic xAI and what can be usefully presented to an end-user, and these "men in the middle" could help to identify needs and solutions. Among the main highlights they found in their interviews there are the following:

- **numerical metrics**, like the confidence of a prediction or the accuracy of a classifier, are **hard to interpret** for many users, they do not know if a certain number means they can trust the system or not. Despite this, an explanation system should try to inform the user about the limitations in the abilities of the AI system.

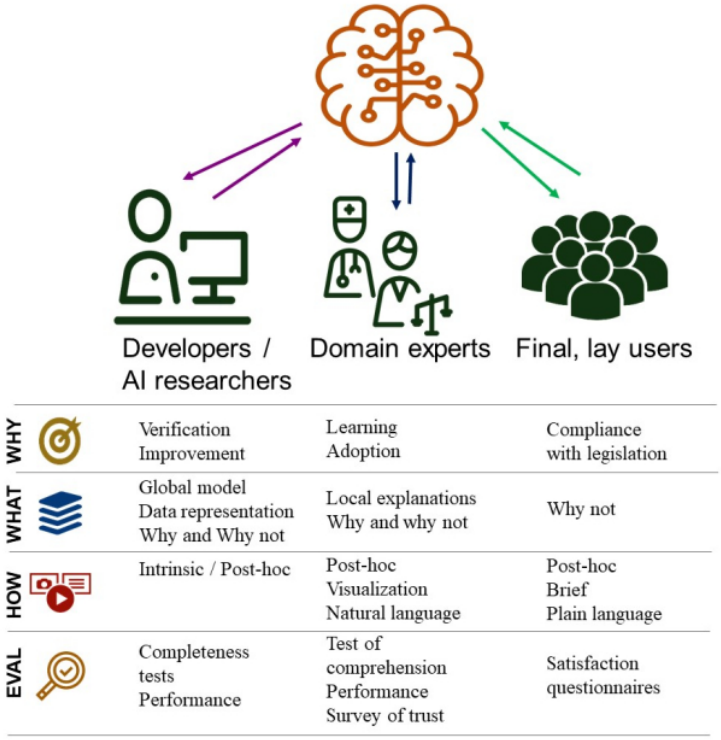

Figure 2.7: User-centric explanations proposed in [36]

- there is the need for a balance between providing the users with enough information to trust the system and **not overwhelming** them
- explanations should be **interactive**, in a kind of conversation where the users can ask for more information depending on their needs and their prior knowledge
- users are not particularly interesting in global explanations about how the system works, they are more interested in **local explanations** for a specific input
- **what-if** and **counterfactuals** explanations can be a valuable tool to help the users

Miller [38] presents an interesting research on xAI characteristics and needs from the social sciences point of view. A first highlight is that explanations must be **constrastive**, i.e. they should not simply tell why something

happened but also why something else did not happen. Even if the users do not explicitly ask for this, they always have it in mind and they appreciate this kind of explanations. We can see a clear reference to the counterfactuals that we presented in Section 2.3.4. Another point is that often the reasons behind a result can be many, but people are used to consider a small subset of them, selected according to certain criteria. These criteria are not uniquely determined, but in general necessary conditions, or necessary and sufficient when they exist, are preferred, even if also the robustness, i.e. the persistency of the result if that factor is kept unchanged despite of the rest, is somehow taken into consideration. Miller also highlights the importance of the way in which explanations are communicated: they should follow a **conversational model**, which does not necessarily mean exactly a chatbot-like conversation, but a way of interacting with them which allows the user to explore and get further information if s/he finds this useful.

The idea of a conversational model is also present in the proposal of Dazeley et al. [39]. They believe that a strong explanation system should be based on a conversation between the system and the users, where the system should first of all identify the type of user that it is interacting with and then start to answer his/her questions with high-level explanations, which can be further detailed with more information until the user seems to be satisfied. While this proposal is certainly interesting, there is no real clue on how this should be implemented, and not even examples of explanations at the different levels or of the different types of users which could be identified by the system.

An interesting analysis on the needs of AI users in the **medical domain** is presented in [40]. The authors analyze the past and the present of medical AI and consider the possible future scenarios, highlighting the high level of disillusion which is present among clinicians when speaking about AI. While the rest of the world emphasizes the recent progresses of AI, clinicians have already seen the results of the first (expert systems) and the second (machine learning) era of AI and they have found them too poor, in the first case, or too opaque, in the second one. Doctors (and patients) cannot use black box models, they need to trust the models and to check their behaviour, understanding the reasons behind it, from both an ethical and a legal point of view. There are only two possible solutions: the use of white-box models,

whose performances have never been particularly good, or the use of post-hoc explanations on black-box models. They argue that AI systems should not be validated only in terms of performance metrics like accuracy or recall, but in terms of *"descriptive accuracy"*, i.e. ability to explain themselves, and *"relevancy"* of their explanations with respect to users' needs.

2.3.6 Textual explanations

There is not much research work related to systems able to produce textual explanations. A position paper about the importance of textual explanations, together with visual explanations, is [41]. Sevastjanova et al. discuss the importance of text as an additional and complementary element to visual explanations, they consider the different ways in which textual explanations could be provided (eg: on demand, on user exploration of certain details, as a repetition of graphically encoded information or as a summary of them or as source of additional information) and they highlight the importance of interactive explanations, possibly including dialogue systems. They also mention counterfactual elements as something which could be easily integrated in textual explanations.

An interesting attempt is the one of Kim et al. [42] which developed a system to produce textual explanations for the behaviour of self-driving cars. Despite the good idea, the system seems more similar to a video captioning system than a real explanation system. In their system the car controller receives as input the video from the car camera and it produces as output the signals which control the vehicle behaviour, together with the computed attention over the video frames, then both the car controller output and input are given as input to the explainer, using two different attention models for the explainer, which are compared in the paper. The main problem of this approach is the training of the explainer: it is trained with data annotated by users, which receive video frames from the car camera and are asked to write a description of the car behaviour and a reason for it. In this way they are basically describing a possible reasonable motivation for a certain car behaviour, but they do not have any information about the reasoning of the self-driving car. There is no evidence that this textual explanations are really correct, and in the human-evaluation procedure is not exactly clear if and how the judges use the attention maps.

Another work on textual explanations has been produced by Hendricks et al. [43] in the context of image classification. They train an LSTM-based model to produce textual explanations for an image classifier, starting with a dataset of bird images, image class labels and image textual descriptions (which are not explanations). In this way the LSTM model produces a text which is actually a description, not really an explanation, but they force this description to highlight the relevant elements which determined the image label by using two particular loss functions for the explainer: a discriminative loss function and a relevance loss function. The latter favors words which are associated to that class in the ground truth of descriptions, while the first one favors sentences which are relevant to discriminate between classes, and so potentially to explain that class. In particular, this is done by using an LSTM-based classifier, which predicts the class on the base of a description, and using its output to compute this loss: a discriminative sentence should produce a correct class label on the LSTM classifier. This LSTM classifier is trained on the ground truth sentences, and this is a first weakness, since these sentences are not necessarily discriminative and in fact it achieves an accuracy of 22%. Another weakness is related to the evaluation. In the automatic evaluation they measure the similarity of an output sentence with the ground truth descriptions of the images belonging to the same class, to assess the class relevance of the sentence. In the human-evaluation, they ask to evaluate the explanations produced by different models, considering how they think they are correct in highlighting the discriminative elements for that class. In both the automatic and the human-based evaluation there is no real assessment of a correspondence between the elements highlighted by the textual explanation and the ones really used by the the image classifier. An explanation should explain the reasons which determine the model output, not the reasons which would determine the "human output", since there is no guarantee of a correspondence between the two. Apart from the evaluation, the explanation generation procedure seems to suffer of the same problem, there is not a strong connection between the classifier reasoning and the explainer training. Another consideration is that this system has been trained and tested on bird images, with classes corresponding to different bird species. In this case there could be (but this should be verified) a link between image description and classifier's output explanation, but this may be much less

true for other datasets, where images contains more elements, which may be absent in a generic description but may be crucial for a classifier.

2.3.7 Interactive explanations

We have found two examples of chatbots for explanations of machine learning models. The first one is a prototype developed by Kuźba and Biecek [44] using the Google's *Dialogflow* framework. They train a classifier on a dataset and they initially train the dialogue system with a small set of (question, intent) pairs. Then they collect additional data through a user study and they retrain it. The system is basically able to provide only three types of interactions:

1. build of a sample, predicting its output
2. breakdown plots in response to feature importance questions
3. ceteris paribus plots in response to what-if questions

The main limitation is the fact that the system answers mainly with plots to users questions, and only with two types of plots. The demo which is currently available online seems to have problems in understanding user's sentences, even simple ones like *My age is 20* or *I am 20 years old*. The main purpose of their work is in fact to discover the users' needs, they observed that the main types of questions asked by the users are why, what-if and feature importance questions, which again confirm the analysis of Section 2.3.5.

Another attempt is the one of Werner [45], which developed a first prototype of a conversational interface called ERIC (Rule-based Interactive Conversational agent for Explainable ai). After having collected the user data it can offer the following types of explanations:

1. textual rules corresponding to the boundaries of the classifier in the feature space
2. SHAP force plots
3. ceteris paribus plots

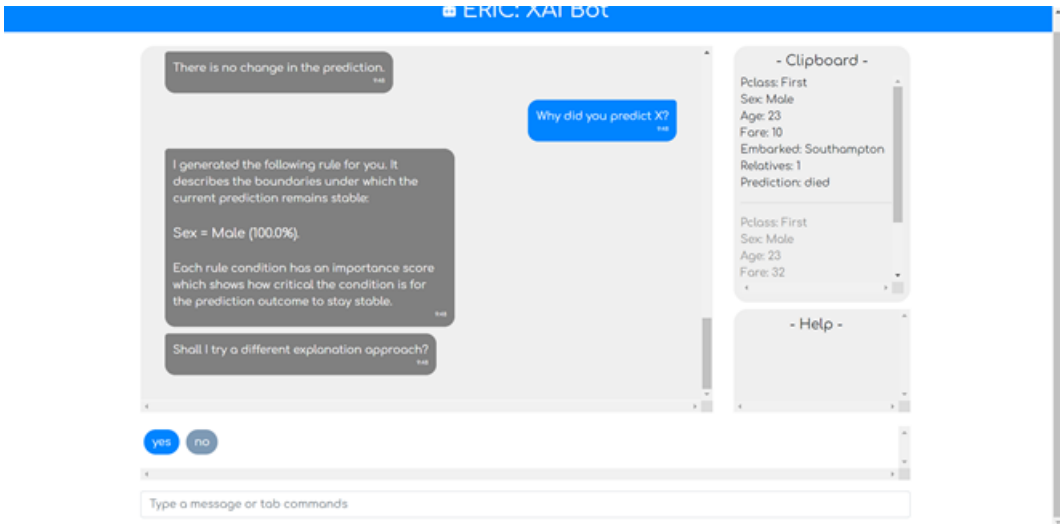

Figure 2.8: Screenshot from the ERIC system

The system computes a similarity between the user sentence and a set of predefined sentences to recognize the rule to apply. It avoids the problem of entity and values recognition through a very rigid series of questions that it asks so that the user has to answer with a specific value (eg: *What is your gender?*) and suggesting to the user the possible valid answers on the interface. A screenshot from the system is in Figure 2.8. The main limitations are the interaction, which is very rigid, and again the explanations that it offers, which are not easy to understand for non-technical users.

2.3.8 Interfaces for xAI

In [46] Liao et al., continuing the work started in [37], propose a question-driven process for the development of an xAI system, considering a real use case scenario of a system for the identification of patients at high risk of adverse events in a hospital. Their development process is driven by the users' questions, traced back to the ones they identified in their previous work. These questions can be grouped in 4 categories:

1. *Why*
2. *How to be that*, i.e. what could change the outcome
3. Information on the *training data*

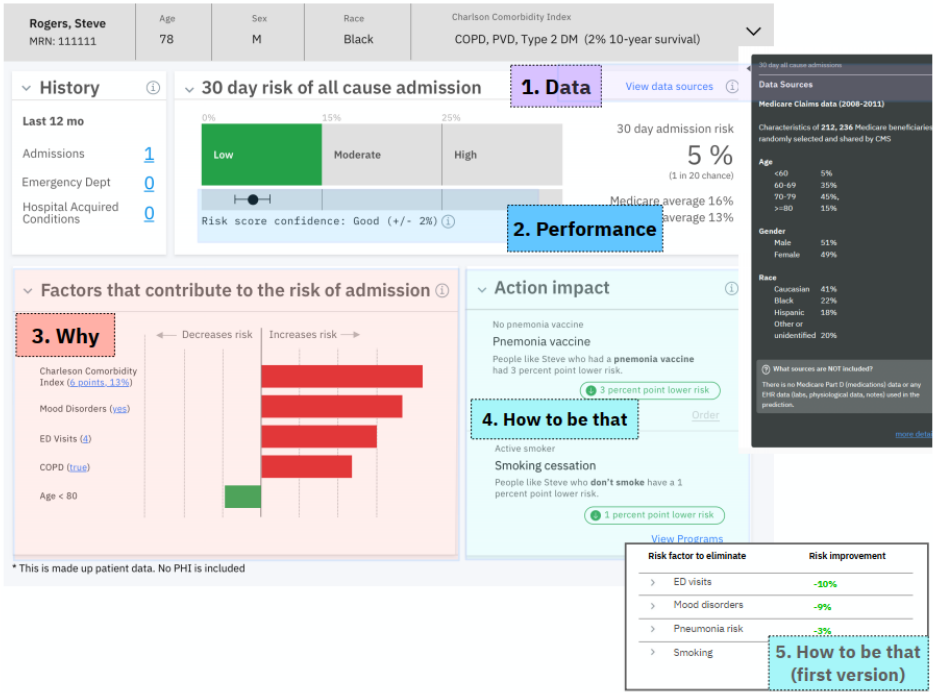

Figure 2.9: The interface shown in [46]

4. Information on *specific conditions* on which the system may have lower performances

The resulting interface is shown in Figure 2.9. They do not detail exactly how the various components are built (eg: which feature importance measure is used?) because the system is proprietary. They collected positive feedbacks about the process from UX Designers, but there is no mention of feedbacks from end users.

The interface that we develop in our work has some elements in common with the one shown in this work, but also important differences (see Section 5.4).

There are various tools and libraries which have been released with the purpose of offering a ready-to-use interface to explain ML models. A first example is *Shapash*[3], whose interface is in Figure 2.10. Another one is *Explainer Dashboard*[4], for which we show a screenshot in Figure 2.11. Both

[3] `https://github.com/MAIF/shapash`
[4] `https://github.com/oegedijk/explainerdashboard`

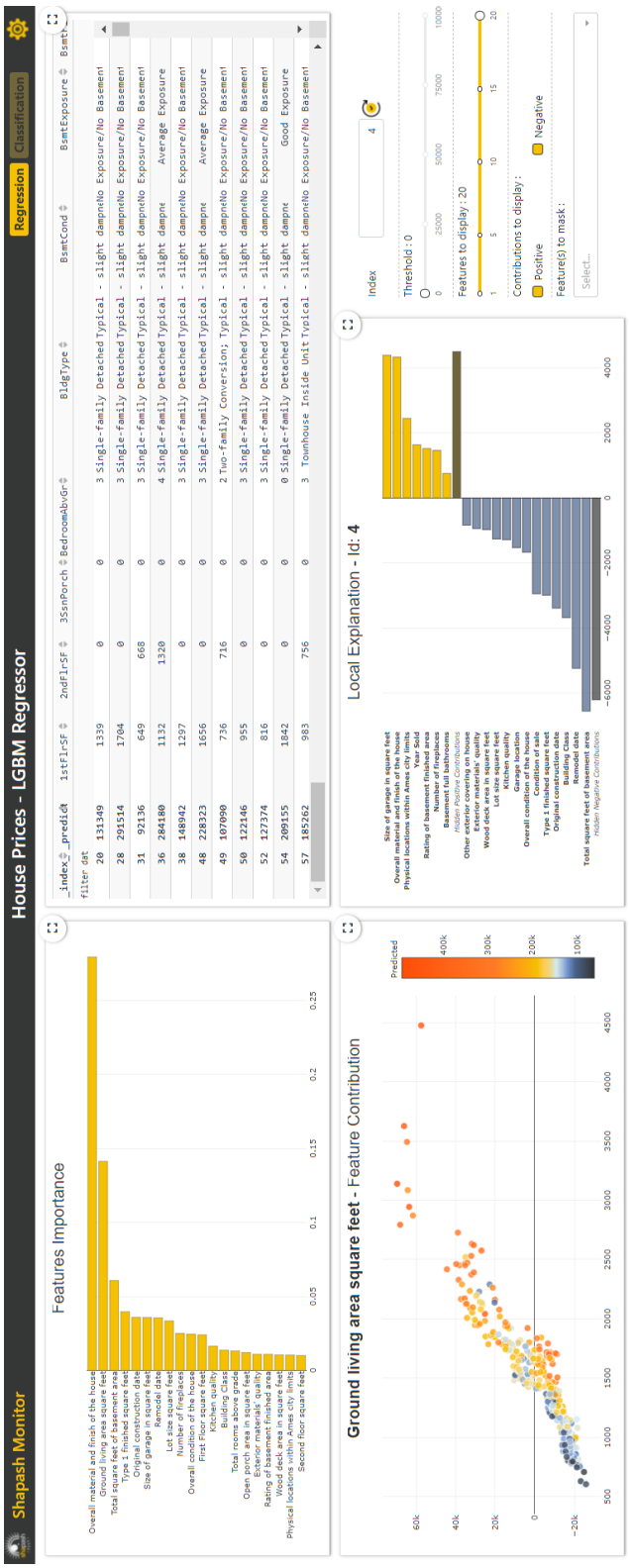

Figure 2.10: The interface of Shapash

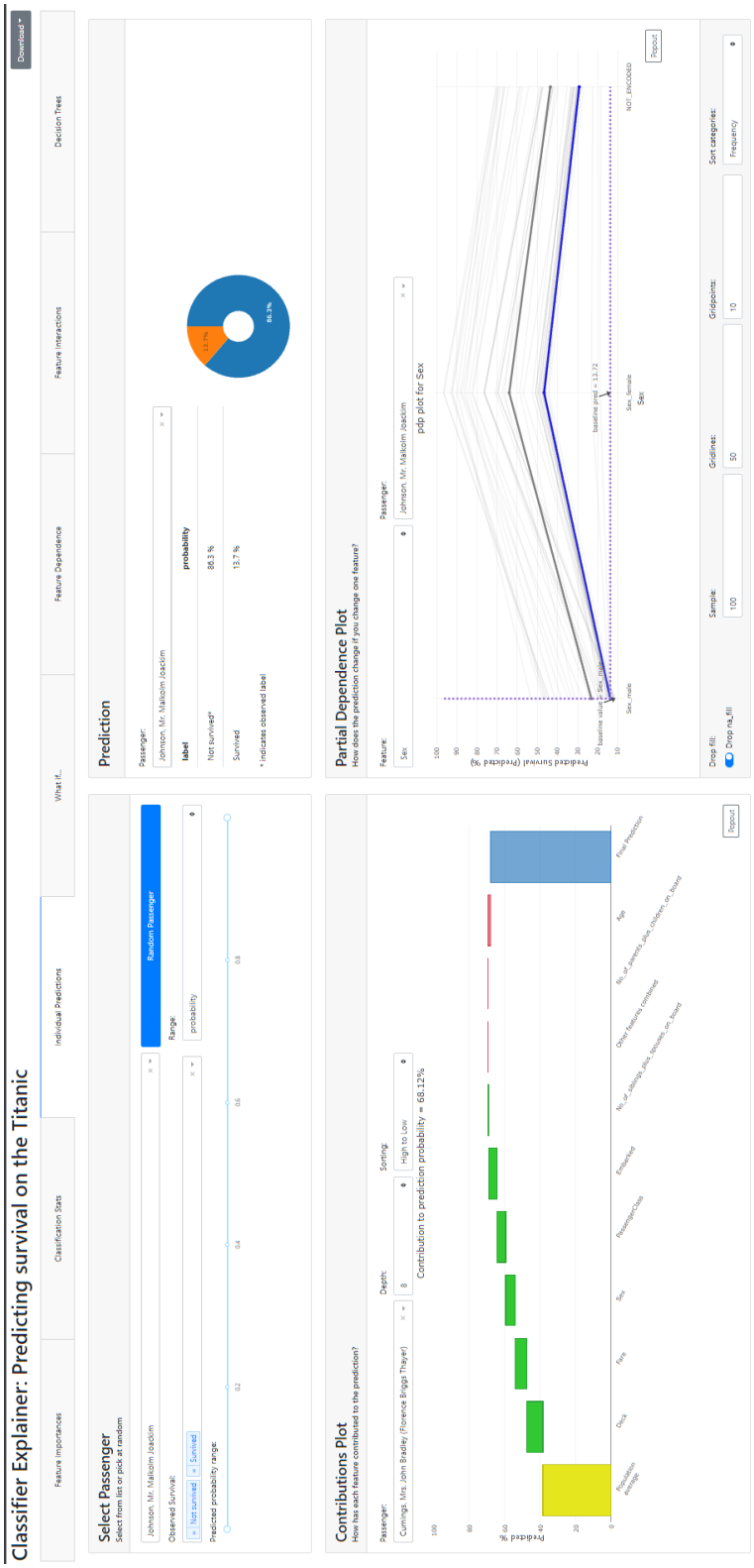

Figure 2.11: The interface of Explainer Dashboard

these two tools can be for sure useful instruments for an xAI practitioner to better understand how a model is working, but they cannot be a tool for an end user. They are basically a collection of plots which can show many different aspects of the model and a non-technical user may be more confused than helped by this kind of visualization. They lack a clear and simple explanation of what is going on during a model prediction. Some of their elements can be useful, but they should be provided to the user in a simpler way, with additional explanations and possibly without showing everything directly, but keeping some elements on demand.

Chapter 3

Research questions

The purpose of this work is to answer the following research questions:

1. *Is it possible to produce textual explanations for a black-box classifier using a generative language model?*

   Many systems are now based on machine/deep learning classifiers, but users need to have an understanding of what they are doing to fully exploit their power. They need to trust these black-box systems and they also need to understand when and why they are making mistakes. The problem of explainability in AI is an active field of research and many techniques have been proposed in the last years, but no one, to the extent of our knowledge, produces an explanation in natural language which can be easily understood by a user, with a generative model like GPT-2. Since its release GPT-2 has been a revolutionary tool in NLP and we believe it has still potential which has not been fully exploited. The construction of a system of this type opens other non trivial questions:

   (a) What should the basis of these textual explanations?

   (b) How can we build a dataset with these explanations to train the generative model?

   (c) What kind of input should be given to the generative model?

2. *How can we evaluate explanations in natural language?*

Given an explanation in natural language, it is non-trivial to define an appropriate evaluation metric. On the one side, correctness, clearness and completeness may seem to be the most relevant properties, but they are hard to measure in an automatic way. On the other side, typical numerical metrics used in NLP may not be able to capture relevant properties.

3. *Is it possible to allow the users to interact with a natural language explainer?*

   A user may not be fully satisfied by a specific explanation, s/he may want to ask something more to understand better the reasons behind the output s/he sees. It seems reasonable that a perfect explanation system would have to allow its users to interact with it.

4. *Which elements of an interface can help users to understand the behaviour of a black-box classifier?*

   There are many elements that can be shown when speaking about a classifier and a dataset. Which ones are useful for a user, especially a non-technical user, to understand better the data and the model? Textual explanations may be a key component, but not necessarily the only one.

Chapter 4

Datasets

4.1 MedDialog dataset

The MedDialog dataset [47] consists of a set of 3.4 million dialogues between patients and doctors in Chinese language and of a set of 260,000 dialogues between patients and doctors in English language. We consider only the English subset, since we are interested in training models which speak English. To our best knowledge it is the largest dataset of medical dialogues in English. This dataset is used for a pre-training of our language model, before training it on the explanations dataset that we automatically generate and augment. In the same way it is used to pre-train the language model for the Q&A system.

4.2 Cardiovascular disease dataset

Our system aims to explain the output of black-box classifiers on datasets with numerical and categorical features. The first dataset that we select for our experiments is a cardiovascular disease dataset freely available on Kaggle[1]. This dataset is originally composed of 70,000 samples, with 11 features and a binary target, described in Table 4.1.

We apply some preprocessing to this dataset:

1. remove samples where systolic blood pressure is outside the range [80-250]

[1] `https://www.kaggle.com/sulianova/cardiovascular-disease-dataset/`

Feature	Type	Categories
ap_lo	Integer	
ap_hi	Integer	
age (days)	Integer	
height (cm)	Integer	
weight (kg)	Floating Point	
gender	Categorical	Male, Female
cholesterol	Categorical	Normal, Above Normal, Well Above Normal
glucose	Categorical	Normal, Above Normal, Well Above Normal
smoke	Categorical	Yes, No
alcohol	Categorical	Yes, No
physical activity	Categorical	Yes, No
cardio	Binary output	Cardiovascular disease, No cardiovascular disease

Table 4.1: Structure of the cardiovascular disease dataset

2. remove samples where diastolic blood pressure is outside the range [40-120]

3. convert age in years (as floating point)

4. replace weight and height with BMI ($BMI = weight(kg)/height(m)^2$)

5. remove samples where BMI is outside the range [5-50]

The ranges which have have been used are quite large on purpose: we wanted to keep all the meaningful values, even if they are outliers, since they could be interesting to explain. We removed only those values which are clearly due to errors in measurements or reporting. The conversion of age in years makes it easier to be understood by a user, without loosing information. The replacement of weight and height with the BMI is done because this is a more reasonable element to determine the risk of cardiovascular disease for a patient than the height or weight alone. This may not be a problem for a good classifier, but again this makes the results easier to be interpreted by a user.

After these operations the dataset contains 68,407 samples, almost perfectly balanced between the two classes.

Feature	Type
Age	Integer
Pregnancies	Integer
Glucose	Integer
Blood Pressure (diastolic)	Integer
Skin Thickness (mm)	Integer
Insulin (U/ml)	Integer
BMI	Decimal
Diabetes Pedigree Function	Decimal
Outcome	Binary output

Table 4.2: Structure of the Pima diabetes dataset

4.3 Pima diabetes dataset

The second dataset that we consider, to understand the generalization capabilities of our models, is the Pima Diabetes dataset[2]. It is again a binary classification dataset, related to the prediction of diabetes. It is much smaller than the cardiovascular disease dataset, containing 768 samples. The data are all related to the Pima indian heritage, from which the name of the dataset. Its structure is described in Table 4.2. The *diabetes pedigree function* is a value related to the familiarity of diabetes, so that the higher it is the higher is the risk of diabetes.

We apply some preprocessing, following the same rationale of the cardio dataset preprocessing:

1. remove samples where insulin is higher than 400
2. remove samples where diastolic blood pressure is outside the range [40-120]
3. remove samples where BMI is higher than 50
4. replace the missing values, present in various columns, with the median of their column

After these operations the dataset contains 737 samples.

[2]https://www.kaggle.com/uciml/pima-indians-diabetes-database

Chapter 5

Approach

In this chapter we explain in detail our approach for the generation of textual explanations and for the Q&A system. In addition, we show our interface, which is also used for the user study presented in the next chapter. In Section 5.1 we show the classification models we have considered. They are the models that we want to explain with our system. In Section 5.2 we detail the different types of textual explanations that we have progressively considered and how we have built a model for their generation, including the construction of the training set and the input encoding. In Section 5.3 we discuss the development of the Q&A model, with the construction of its training set and its input and output formats. Finally, in Section 5.4 we show our interface, discussing the main elements which compose it.

5.1 Classifiers

We use different classifiers, all treated as black-box models by our explanation system, to demonstrate that our approach works independently from the underlying model. In particular, we consider the following classifiers:

- **XGBoost**
- **Random Forest**
- **Logistic Regression**
- A **Feed Forward Neural Network** with 2 hidden layers and a sigmoidal output

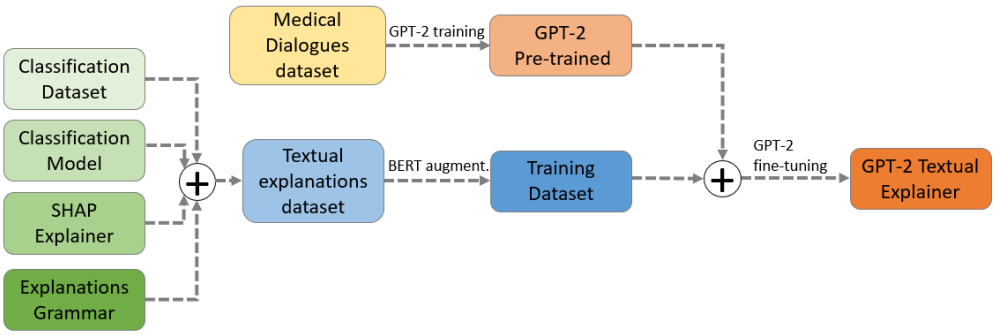

Figure 5.1: Schema of the architecture to train the textual explanations model

Their results are compared in Chapter 6, but we consider them all since the focus of this work is the ability to explain them, even when they are wrong, so that a user can understand it. Our system can work with any classifier, in particular our demo can work with any model respecting the Scikit-learn interface[1]. For neural networks there are wrappers like the *KerasClassifier* that we have used.

5.2 Explanations

A schema of the architecture used in the training of our explanation model is in Figure 5.1. The main purpose of our system is to produce textual explanations with a generative language model and this requires a dataset of samples and associated textual explanations to train the language model. A dataset of this type does not exist and this is the first challenge we have to face.

The first point to consider is how to define the textual explanations. What should they contain? On what basis should they be formulated? There are different possibilities, related to the the research that has been done so far in the field of xAI. Initially, in the first version of our explainer, we decide to focus on simple explanations which show to the user which are the most important features that determine the output for the sample. This means that the textual explanations have to be based on some measure of the importance of the features for the specific sample. Which measure should we

[1] `https://scikit-learn.org/stable/developers/develop.html`

use? There are again different possibilities, focusing on the model-agnostic ones we can consider LIME , SHAP or the approach of Baehrens et al. [28]. We decide to use SHAP because of its solid theoretical guarantees, already mentioned in Section 2.3.3. We use the *Dalex* python library [48] to compute the Shapley values. Among the various methods to compute the Shapley values they use an approximation similar to the one of [30].

For a given sample and a classifier we compute the Shapley values, then we consider the three features with the highest positive Shapley values (in rare cases there may be only two or one) as a base for our textual explanations. After the first basic version, we expand it progressively, including more information:

- **Version 1**: only the most relevant features

  eg: *The prediction of disease is determined by the systolic blood pressure (140) and by fact that he is a smoker. The BMI (29.3) also contributes to the result.*

- **Version 2**: additional information on mean and standard deviation of numerical features

  eg: *The prediction of disease is determined by the systolic blood pressure (140), which is one standard deviation above the mean, and by the fact that he is a smoker. The BMI (29.3), higher than the mean, also contributes to the result.*

- **Version 3**: includes also the description of a counterfactual

  eg: *The prediction of disease is determined by the systolic blood pressure (140), which is one standard deviation above the mean, and by the fact that he is a smoker. The BMI (29.3), higher than the mean, also contributes to the result. If BMI was 27 and systolic blood pressure was 130, then the prediction would be no disease.*

- **Version 4**: includes also an explanation related to the information which comes from the ceteris paribus plot (see Section 5.4.3), a plot showing how the result changes when we change only a certain feature

  eg: *The prediction of disease is determined by the systolic blood pressure (140), which is one standard deviation above the mean and whose high*

values are associated with cardiovascular disease, and by the fact that he is a smoker. The BMI (29.3), which is higher than the mean and whose high values increase the likelihood of cardiovascular disease, also contributes to the result. If BMI was 27 and systolic blood pressure was 130, then the prediction would be no disease.

5.2.1 A grammar for automatic generation

To build the training set for our generative language model we need to define a grammar and some rules to compose explanations in an automatic way. We start with a basic grammar for the first version of the explanations, which is presented below, with $\{f\}$ as a placeholder for the feature name and $\{v\}$ as a placeholder for the feature value:

S → *F SN T*
F → *The main reason why P has been predicted as O1 is the EF*
| *The first element which influenced the prediction of O2 is the EF*
| *The most relevant factor for the prediction of O2 is the EF*
SN → *. In addition, the EF also has a significant influence*
| *. The EF is also an important element*
| *. Moreover, the EF plays an important role*
T → *and also the EF is relevant .*
| *, while the third factor is the EF.*
| *. Finally, the EF also influences the result .*
P → *he* | *she*
O1 → *having no cardiovascular disease* | *having a cardiovascular disease*
O2 → *no cardiovascular disease* | *cardiovascular disease*
EF → *AGE* | *OTHER_NUM* | *PA* | *ALCOHOL* | *GENDER* | *SMOKING* | *GLUCOSE* | *CHOLESTEROL*
AGE → *fact that patient is elderly* | *fact that patient is young* | *fact that the patient is middle−aged*
PA → *physical activity of the patient* | *inactivity of the patient*

ALCOHOL →*use of alcohol* | *absence of use of alcohol*
SMOKING →*fact that the patient is a smoker* | *fact that the patient is not a smoker*
GLUCOSE →{*v*} *level of glucose*
CHOLESTEROL →{*v*} *level of cholesterol*
OTHER_NUM →*value of* {*f*} *(*{*v*}*)*

Here the rules to select the production to apply in the various non-terminal symbols are straightforward, based on the type of output, on the feature name or on the feature value. For the first non-terminals instead there is no fixed rule to follow, the selection is random.

For the second version of the explanations the previous grammar is expanded in the following way, where {*n_std*} corresponds to the number of standard deviations that the value is above or below the mean:

OTHER_NUM →*value of* {*f*} *(*{*v*}*) DIST*
DIST →*, which is* {*n_std*} *standard deviations above the mean,*
| *, which is* {*n_std*} *standard deviations below the mean,*
| *, which is higher than the mean,*
| *, which is lower than the mean,*

The third version requires a further expansion to include the information about the counterfactual:

S → *F SN T CF*
CF →*FIRST_CF_F CF_F*
FIRST_CF_F →*If CF_F_DESC*
CF_F →*, CF_F_DESC* | ϵ
CF_F_DESC →*ALCOHOL_CF* | *SMOKE_CF* | *GENDER_CF* | *OTHER_CF*
ALCOHOL_CF →*the patient drank alcohol* | *the patient did not drink alcohol*
SMOKE_CF →*the patient was a smoker* | *the patient was not a smoker*
GENDER_CF →*the patient was a male* | *the patient was a female*
OTHER_CF →{*f*} *was* {*v*}

Here only the features which are changed are shown, and in particular for numerical features we highlight only the changes higher than 5% of the

range, since very small changes may be irrelevant and due to the fact that there is not another sample in the dataset which keeps the values precisely identical.

For the last version we need to add a non terminal *WHY* after the description of each feature, which has the following productions:

WHY →, *where high values of this attribute are associated with a high probability of cardiovascular disease* |
, *where high values of this attribute are associated with a low probability of cardiovascular disease* |
, *where low values of this attribute are associated with a high probability of cardiovascular disease* |
| , *where low values of this attribute are associated with a low probability of cardiovascular disease*,
| ϵ

In particular, to determine which of the above descriptions has to be used, we pick the following points from the ceteris paribus plot: (min + range/6), (min+2*range/6), (min - range/6), (min - 2*range/6). If the value is higher than the mean we consider the last two points (*high values*), otherwise we consider the first two (l*ow values*). If the value in the considered points is higher than 0.7 we say that they are associated with *high probability*, if instead it is lower than 0.4 we say they are associated with *low probability*.

At the end we consider also an extended version of the grammar, which adds more variability to the text:

S → *F SN T CF*
F → *The main reason why P has been predicted as O1 is the EF*
| *The first element which influenced the prediction of O2 is the EF*
| *The most relevant factor for the prediction of O2 is the EF*
| *The first motivation for the prediction of O2 is the EF*
| *The O2 outcome is primarily determined by the EF*
| *The first cause determining the outcome of O2 is the EF*

| *The EF is a significant factor which determines the outcome of O2*

| *Of primary importance for predicting O2 is the EF*

| *The EF is important for predicting O2*

| *The diagnosis of O2 for this patient is based on the EF*

$SN \rightarrow$ *. In addition, the EF also has a significant influence*

| *. The EF is also an important element*

| *. Moreover, the EF plays an important role*

| *. The second important element is the EF*

| *. Another important feature is the EF*

| *. Furthermore, the EF has a considerable effect*

| *. A second factor to consider is the EF*

| *. The EF also has a significant effect*

| *. The EF is another major factor*

$T \rightarrow$ *and also the EF is relevant .*

| *, while the third factor is the EF.*

| *. Finally, the EF also influences the result .*

| *and the EF also affects the prediction*

| *and the EF also contributes to the result*

| *. The result is also affected by the EF*

| *. The EF is third factor which determines the outcome*

| *. It is important to mention the EF as well*

| *. The EF is also worth mentioning among the causes*

5.2.2 Generative language model

For the generative language model we use the well known GPT-2 model, a transformer-based model which has been widely used in NLP since its release. Considering that we want to focus on the medical domain we start to train it on the MedDialogue-EN dataset, so to make it used to the medical language.

5.2.3 Input encoding

In Figure 5.2 we show the schema of the procedure with which our model produces the textual explanation for a given sample, on a given classifier. Given a sample and the related information, we need to encode it in some

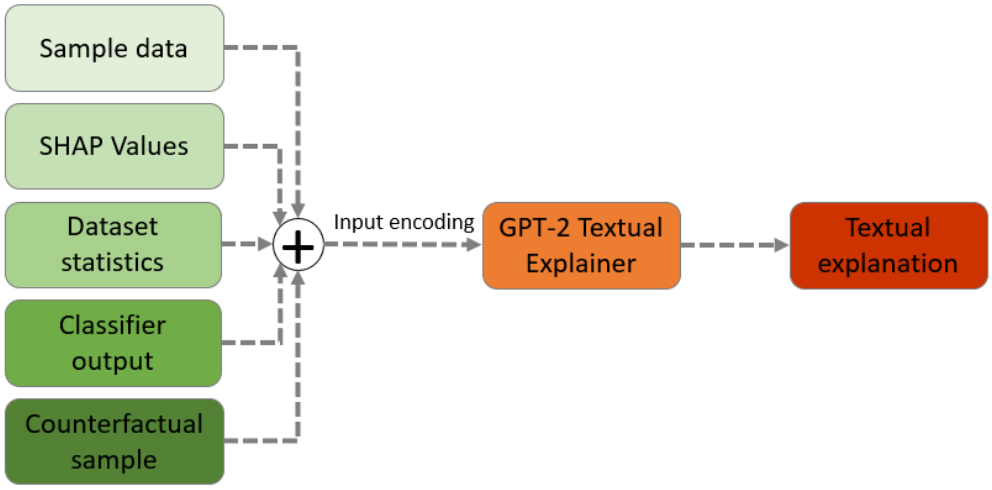

Figure 5.2: Schema of the procedure followed by our system to produce a textual explanation

way. The input for the the GPT-2 model during the training is of the form:

encoded sample <END> textual explanation <|endoftext|>

while at inference time it receives an input of the form

encoded sample <END>

and it has to produce the corresponding textual explanation. There are many possible ways in which we can encode the information which is needed to produce the textual explanation, we consider different possibilities for the first version of the explanations and we keep the one which performs better. The following are the four candidates, shown via examples:

- **Encoding 0**: *age=52; gender=Male; ap_hi=100; ap_lo=70; cholesterol=Normal; gluc=Well Above Normal; smoke=No; alco=No; active=Yes; BMI=40.1; pred=no disease; firstF=ap_hi; secondF=ap_lo; thirdF=age;*

  where at the end we report the three most important features according to the Shapley values

- **Encoding 1**: *input=[age=52; gender=male; ap_hi=100; ap_lo=70; cholesterol=normal; gluc=well above normal; smoke=no; alco=no; ac-*

tive=yes; BMI=40.1], prediction=no disease, explanation=[ap_hi∼0.1; ap_lo∼0.2; age∼0.1]

where at the end we report the three most important features, together with their Shapley values

- **Encoding 2**: *input=[name=age, value=52, shap=0.1]; [name=gender, value=male, shap=-0.0]; [name=ap_hi, value=100, shap=0.2]; [name=ap_lo, value=70, shap=0.0]; [name=cholesterol, value=normal, shap=0.0]; [name=gluc, value=well above normal, shap=0.0]; [name=smoke, value=no, shap=-0.0]; [name=alco, value=no, shap=0.0]; [name=active, value=yes, shap=0.0]; [name=BMI, value=40,1, shap=0.0]; prediction=no disease; explanation=[ap_hi; ap_lo; age]*

- **Encoding 3**: *input=[name=age, value=52, shap=0.1]; [name=gender, value=male, shap=-0.0]; [name=ap_hi, value=100, shap=0.2]; [name=ap_lo, value=70, shap=0.0];[name=cholesterol, value=1, shap=0.0]; [name=gluc, value=3, shap=-0.0]; [name=smoke, value=false, shap=0.0]; [name=alco, value=false, shap=0.0]; [name=active, value=true, shap=0.0]; [name=BMI, value=40,1, shap=0.0]; prediction=no disease;*

  which is the same of *Encoding 2* but without providing explicitly the three most important features

We compare the results with the different encodings (see Chapter 6) and then we keep *Encoding 3*.
For the second version of the explanations we need to encode also the information about mean and standard deviation. We consider two alternatives:

- **Encoding 3.1**

  [name=BMI, value=32, shap=0.1, ***mean=25, std=3.2****]* where we include the mean and standard deviation for every numerical feature

- **Encoding 3.2**

  [name=BMI, value=32, shap=0.1, ***mean=higher, std=2****]* where we include an indication of the fact the sample is below or above or near the mean and we include directly the number of standard deviations it is above/below the mean, for every numerical feature

The results of the experiments make us choose the first one.

In the third version we need to encode also the counterfactual (see Section 5.2.4 for details on its computation), so we modify *Encoding 3.1* by adding a description of the relevant changes that are present in the counterfactual, together with the counterfactual class name:

input=[name=age, value=52, shap=0.1]; [name=gender, value=male, shap=-0.0]; [name=ap_hi, value=100, shap=0.2]; [name=ap_lo, value=70, shap=0.0]; [name=cholesterol, value=1, shap=0.0]; [name=gluc, value=3, shap=-0.0]; [name=smoke, value=false, shap=-0.0]; [name=alco, value=false, shap=-0.0];[name=active, value=true, shap=0.0]; [name=BMI, value=40,1, shap=-0.0]; prediction=no disease; ***cf=[name=age, value=55][name=BMI, value=37.2]; cf_pred=cardiovascular disease***

For the last version we need to encode somehow the information of the *ceteris paribus* plot. We consider two possible variations of the input encoding of numerical features:

- **Encoding 3.1.1**

  [name=age, value=0.52, shap=0.1, mean=40, std=3.5, ***cp_1=0.3, cp_2=0.35, cp_3=0.6, cp_4=0.8****]*

  where we explicitly provide the 4 y-values of the plot, corresponding to x=$[(min + range/6), (min + 2 * range/6), (max - range/6), (max - 2 * range/6)]$

- **Encoding 3.1.2**

  [name=age, value=0.52, shap=0.1, mean=40, std=3.5, ***cp_low=low, cp_high=no****]*

  where we add two words, one for low values and one for high values, which can assume the values *low, high or no*, where *no* means that there is not a clear behaviour

At the end we keep the second version.

5.2.4 Counterfactuals

For the computation of counterfactuals we base mainly on the approach of Wachter et al. [31], i.e. the idea of using a certain distance measure,

normalized by MAD, searching in the neighbourhood of our current sample, so to be sure to produce realistic counterfactuals.

We consider three different possible distance measures:

1. **Manhattan** distance, the one suggested in [31]: $d(x, y) = \sum_i |x_i - y_i|$
2. **Euclidean** distance $d(x, y) = \sum_i (x_i - y_i)^2$
3. **Chebyschev** distance, also known as *max-norm* or *infinity-norm* $d(x, y) = \max_i |x_i - y_i|$

They can be all reasonable distance measures, the Manhattan distance has the well-known property of minimizing the number of altered features, which is particularly useful in the context of counterfactuals, but also the Chebyschev distance could be interesting to avoid points which are too far in a specific direction (i.e. feature), even if that is the only different feature (or one of the few different ones), and we consider also the classic Euclidean distance to assess the difference with the other two.

In our interface we show three counterfactuals to the users, one for each of the above distance measures. The information provided by a single counterfactual may in fact be not enough to have a full understanding of the behaviour of the classifier. For instance, if a feature is changed in a counterfactual we may consider that feature important for the classification of the current sample, but this importance is always relative to the other feature values, it may be that a change in two other features without a change in this feature can lead to a different counterfactual, not necessarily worse. Nevertheless, it may happen that different distance measures lead to the same counterfactuals. An example of a sample with its counterfactuals is in Table 5.1.

We performed an analysis in term of number of changed features, over a random subset of 1000 samples of the cardiovascular disease dataset, the result is in Table 5.2 .

We can notice how the Manhattan distance is the one which determines less feature changes, but the Euclidean distance is not far from it. Anyway, as we have already mentioned, this is not necessarily the best metric to evaluate counterfactuals and because of this we keep all the three distance measures in our interface, while when we have to consider a counterfactual for the textual explanations we use the one computed with the Manhattan distance.

Metric	Age	BMI	Gender	BP High	BP Low	Cholest.	Glucose	Smoke	Alco.	P.A.	Prediction
	46	32.0	Female	110	70	Normal	Normal	No	No	No	No Disease
Scaled Manhattan	**50**	**33.1**	Female	**100**	70	Normal	Normal	No	No	No	Disease
Scaled Euclidean	**45**	**29.0**	Female	110	70	**Well Above Normal**	Normal	No	No	**Yes**	Disease
Scaled Chebyschev	**47**	**32.8**	**Male**	110	70	**Above Normal**	**Above Normal**	No	No	No	Disease

Table 5.1: Example of a sample with counterfactuals computed with different distance metrics

Metric	Changed Features	Changed feat. (change $> 5\%$)
Scaled Manhattan	1712	410
Scaled Euclidean	1823	453
Scaled Chebyschev	2609	1069

Table 5.2: Total number of changed features on the counterfactuals for 1000 samples from the cardiovascular disease dataset

5.2.5 Augmentation

We consider different augmentation techniques to increase the variability of the training set. A first possibility is a synonym replacement using dictionaries or word embeddings, but a better technique is to use a language model like BERT [17] which is able to predict a masked word in a sentence, taking into account the entire sentence. Another technique which can be used to augment text is the so-called *Round Trip Translation (RTT)*, where we translate the text into a different language and then we translate it back to the original language. RTT present some problems, first of all the limitations of the translation APIs, but also the fact that often it does not lead to any augmentation, producing the exact same sentence from which it started. Its results depend on the choosen languages, but also on the number of steps in translation chain, the more they are the more likely it is to observe a difference, but every step is an additional cost. Because of these reasons we focus on augmentation with BERT.

In particular, for each explanation we mask a single word, excluding the stop words present in the *nltk English stop words dictionary*, the name of the features, which we want to keep unchanged to be sure that the user un-

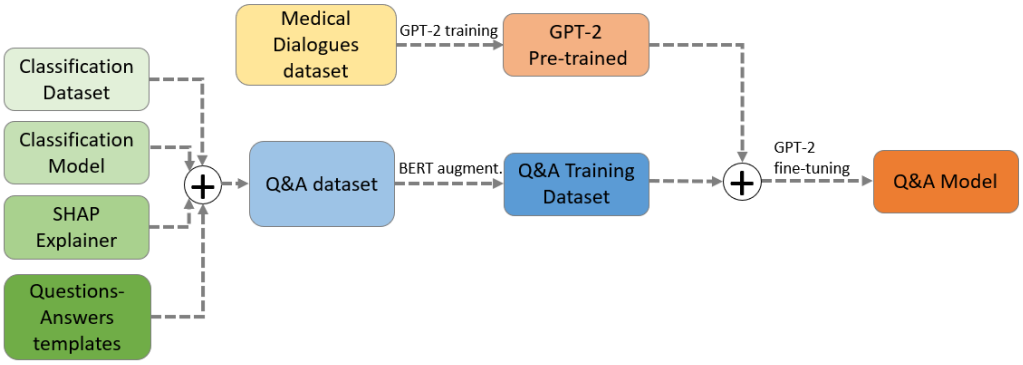

Figure 5.3: Schema of the architecture to train the Q&A model

derstands what we are referring to, and the numerical values of the features. Then the masked words are replaced with the prediction proposed by BERT. This operation can also be repeated multiple times. We experiment with it up to three rounds.

5.3 Question answering system (Q&A)

We would like to offer to the users the possibility of an interaction with the explainer, which seems to be a common request according to the researches discussed in Section 2.3.5. The idea is to offer the possibility of asking some questions to the system. We continue to use a generative model for the Q&A system, whose training architecture is similar to the one we have used for the explanations. We show it in in Figure 5.3, while in Figure 5.4 we show the schema of the procedure used by our system to produce an answer for a question on a given sample, on a given classifier.

We start to consider questions like *How important is age?*, *What is the importance of BMI?*, related to the importance of the various features.

We build a training set with a (question, answer) pair for each sample of the cardiovascular disease dataset. We consider the following questions, where $\{f\}$ is the name of a feature:

1. *What is the importance of* $\{f\}$*?*

2. *How about* $\{f\}$*?*

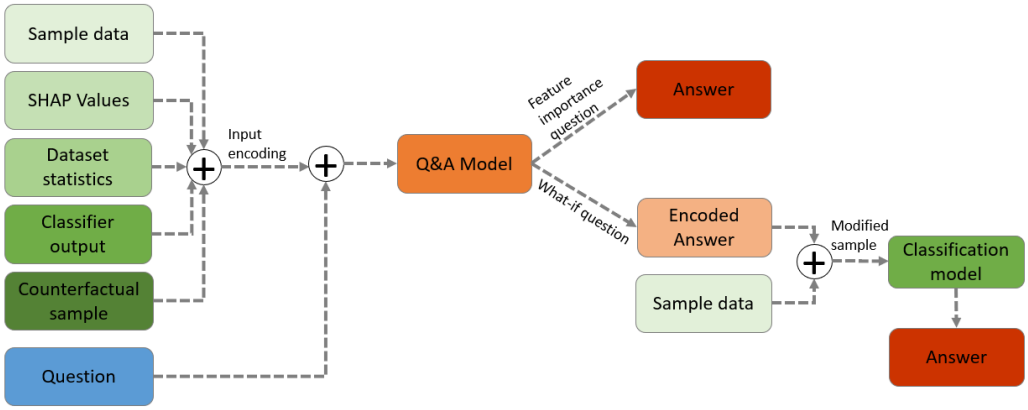

Figure 5.4: Schema of the procedure followed by our system to produce an answer

3. *Is {f} relevant?*
4. *How important is {f}?*
5. *What can you say about {f}?*
6. *How does {f} affect the prediction?*

For each sample we choose a feature randomly, we select randomly one of the questions and we determine the type of the answer on the base of the Shapley value of that feature for that sample: positive, negative or neutral. Given the type of the answer, we randomly choose it from the set of answers of the corresponding group.

Positive answers:

1. *{f} is relevant for the prediction*
2. *{f} has a positive contribution to the prediction*
3. *{f} is one of the reasons for the result*

Neutral answers:

1. *{f} has a negligible effect on the prediction*
2. *{f} has very low influence on the result*

3. *{f} is not relevant for the prediction*

Negative answers:

1. *{f} has a negative contribution to the result*

2. *{f} leads to the opposite prediction*

3. *{f} would be a reason for the opposite result*

This dataset of questions and answers is augmented with the same procedure used for the textual explanations.

The training set for the GPT-2 model is composed of elements of the following format:

encoded input; question=How important is age? <END> answer <|endoftext|>

The second type of questions that we consider are *what-if* questions, like *What if age was 60?* or *What would change if BMI was 25?*. Again we consider this type of questions because they have been mentioned multiple times in the works about the users' needs in term of xAI.

Similarly as before, we define a set of question templates, for each sample we create a (question, answer) pair drawing a question template and filling it with a feature name and with a value, drawn from the distribution of the values for that feature in the dataset. This is the set of template questions that we consider:

1. *What if {f} was {v}?*

2. *What would change if {f} was {v}?*

3. *How would it be if {f} was {v}?*

4. *Would the result change if {f} was {v}?*

5. *What would be the prediction if {f} was {v}?*

The difference is in the answers: we cannot directly express the answers, we do not expect that the GPT-2 model becomes able to predict what would happen by changing a particular value of a sample in a certain way. Instead we want to make the language model able to recognize the feature and the value to change, producing an answer with the following format:

$<WHAT\_IF>\{f\}=\{v\}<|endoftext|>$

For this purpose we explicitly add the $<WHAT\_IF>$ token to the GPT-2 tokenizer. Given this output we can parse it, modify the sample, call the classifier model on the modified sample and produce a textual answer containing the classifier output. Initially we develop and test two separate models for the two types of questions, then we develop a single model able to answer both types of questions. Details on experiments and results are in the next chapter.

5.4 Interface

We develop an interface to show to the users our textual explanations, together with the Q&A and other useful information. It is a web interface, with a Python Flask backend.

In developing the interface we proceed driven by the needs and considerations expressed in Sections 2.3.5 and 2.3.8. We want to build an interface which is clear and complete for an end-user of a classification system. At the same time, this interface must allow us to collect feedbacks about the textual explanations and the Q&A system. In the interface a single sample is shown, together with the prediction of the model and its confidence. For our user-study we allow the users to select between two datasets and different classification models for each dataset. A screenshot of the interface with its main elements is shown in Figure 5.5.

In the top of the interface there is a table which shows the feature values and the model output. Hovering with the mouse (or tapping in mobile devices) on the feature values the user can see the distribution plots and the *ceteris paribus* plots, which are discussed in the following sections. In the bottom there are the *feature importance* plot and the textual explanations, together with the form which allows to collect the user evaluation and the form which allows to try the Q&A system.

5.4.1 Feature importance plot

The feature importance plot is a bar plot showing the Shapley values of the features, highlighting their positive or negative contribution. The bar plot

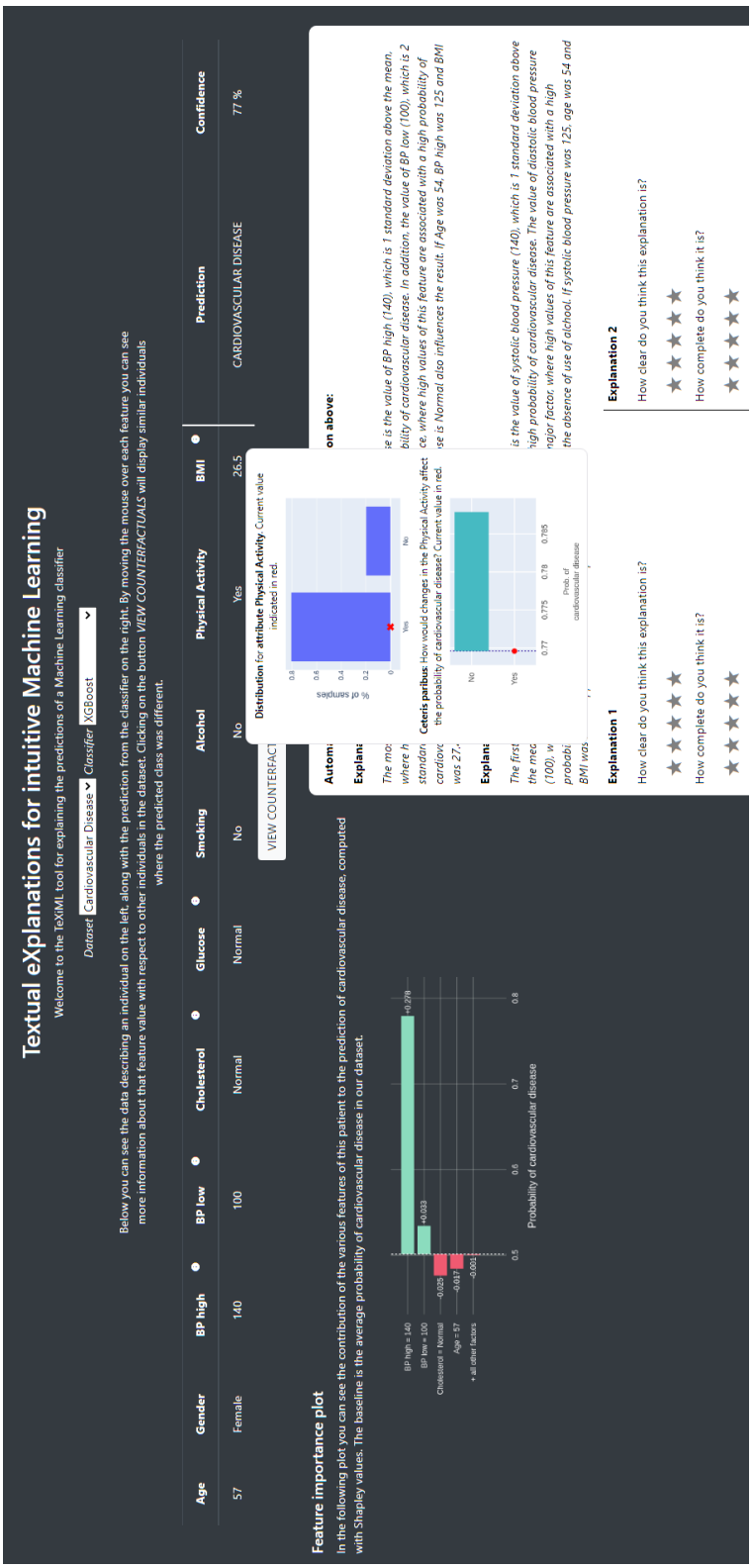

Figure 5.5: Screenshot of the interface with its main elements

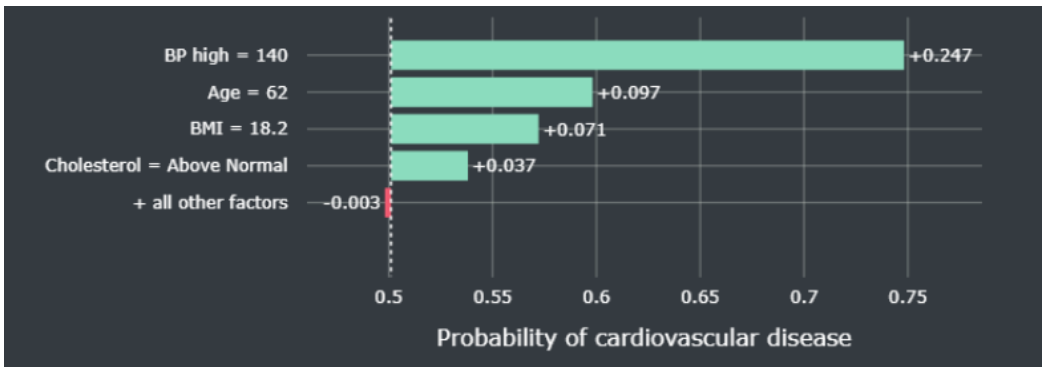

Figure 5.6: Feature importance plot for a sample of the cardiovascular disease dataset

shows in detail values of the first four more relevant features, i.e. the ones with highest absolute values of Shapley values, and then it groups the effect of the remaining ones in a single bar. The starting point of the bars, on the x-axis, is the average probability output over the entire dataset. An example of feature importance plot for the cardiovascular disease dataset is in Figure 5.6.

5.4.2 Distribution plots

Distribution plots are shown when the user moves over a feature name or value. Their aspect is different depending on the type of feature:

- **numerical features**: the plot is an histogram with a KDE (kernel density estimation). The height of the bins is the probability density. The KDE is the *Scipy*'s gaussian KDE, where the bandwith is choosen according to the Scott's rule [49]: $n^{-1/(d+4)}$ where n is the number of data points and d is the number of dimensions (in our case always one). The width of a bin is computed as $2 * \sigma/(n^{0.25})$, which is the formula used as default by the *Plotly* library, which is a slight variation of Scott's rule [50]. The value of the current sample is highlighted by a red cross. An example of distribution plot for a numerical feature is in Figure 5.7.

- **categorical features**: the plot is an histogram with a bin for each category, with the height representing the probability distribution. The

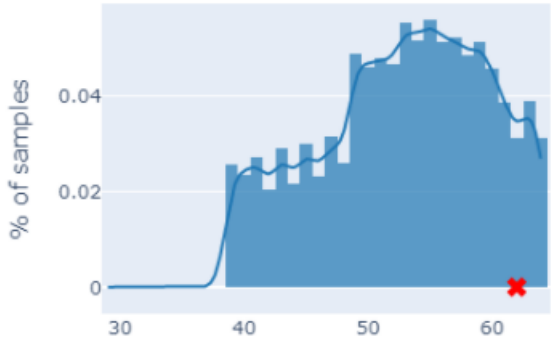

Figure 5.7: An example of distribution plot for a numerical feature, in this case Age in the cardiovascular disease dataset

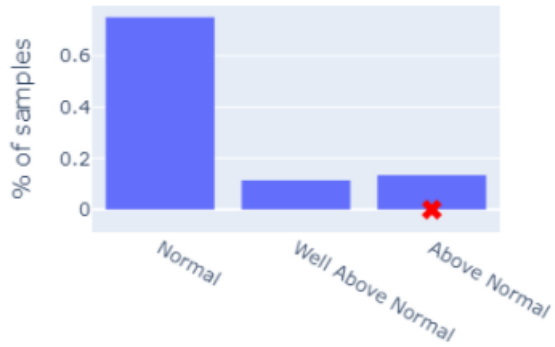

Figure 5.8: An example of distribution plot for a categorical feature, in this case Cholesterol in the cardiovascular disease dataset

value of the current sample is highlighted by a red cross. An example of distribution plot for a categorical feature is in Figure 5.8.

5.4.3 Ceteris Paribus plots

Ceteris paribus plots, or *what-if* plots, show how the model outcome changes for a specific sample when a certain feature is changed, keeping all the other values unchanged. They are illustrated by Biecek and Burzykowski in [51]. We initially use their *Dalex* python library [48] to compute them, but we obeserve that the result obtained in this way is not optimal for numerical features and we modify their implementation. In the case of numerical features the plot is a line which shows the model outcome on the y-axis while the value of the selected feature changes on the x-axis. In particular the

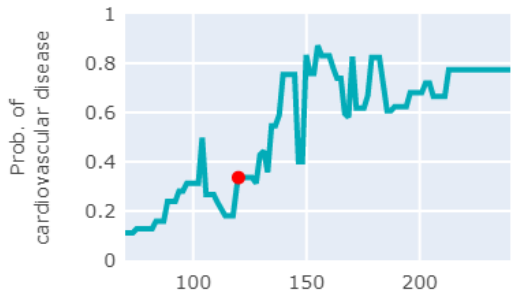

Figure 5.9: Ceteris paribus plot, in the version computed by the Dalex library, for the BP High of a sample of the cardiovascular disease dataset, with XGBoost classifier

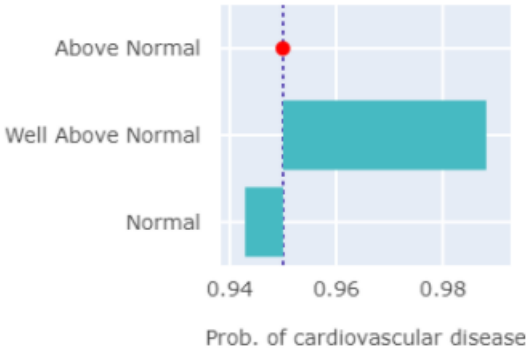

Figure 5.10: Ceteris paribus plot for the Cholesterol of a sample of the cardiovascular disease dataset

values to consider on x-axis for the calculation of the profile are 100, taken uniformly in the range. An example of a ceteris paribus plot, in the version computed by the Dalex library, for a numerical variable, is in Figure 5.9. In the case of a categorical feature the plot is a bar plot, with a bar for each category, showing how the prediction is altered if the value of the feature is changed in that way. An example of ceteris paribus plot for a categorical variable is reported in Figure 5.10.

The plot shown in Figure 5.9 highlights the problem with numerical features: it is very irregular and it can easily confuse a user. This type of result is common with ensamble models, while it is clearly not the case of logistic regression, but even neural networks have a more regular behaviour. Figure 5.11 shows examples of ceteris paribus plots for logistic regression and neural network classifiers. To avoid to show to the users these irregular

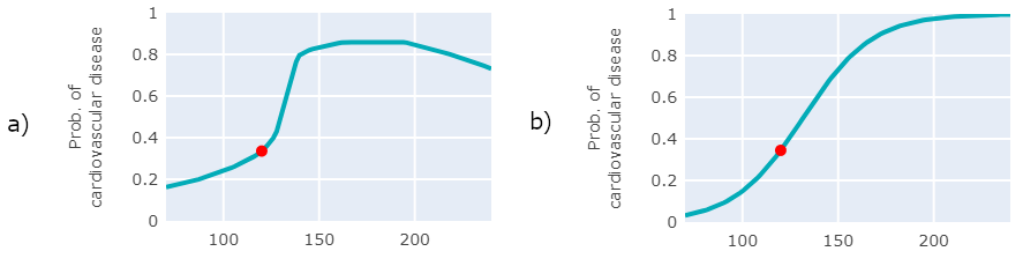

Figure 5.11: Ceteris paribus plots, in the version computed by the Dalex library, for the BP High of a sample of the cardiovascular disease dataset, with a) neural network classifier, b) logistic regression classifier

plots which may be confusing, we adopt a smoothing procedure based on LOESS [52]. LOESS is a local regression technique, which can be used to smooth plots. It fits a local polynomial at each point, using weighted least squares, where weights are inversely proportional to the distance from the point. This weighted least square is also repeated for many iterations, adjusting the weights on the base on the residuals, so to ensure more robustness. LOESS is typically used with polynomial degree equal to 1 or 2. We try both and we decide to use a degree of 2, since it produces smoother plots. A comparison of an original ceteris paribus plot together with the smoothed versions with degree 1 and 2 is in Figure 5.12. To compute the smoothed plot we use the *LOESS_1D* routine of Cappellari et al. [53], which implements the univariate robust LOESS algorithm of Cleveland [52], but we extend it to compute also a confidence interval around the smoothed plot. We compute the pointwise confidence interval with the usual formula:

$$\hat{f}(x_0) \pm z_{\alpha/2} \cdot \hat{\sigma}(l(x_0))$$

where $\hat{f}(x_0)$ is the result of the robust LOESS fitting in x_0, $\hat{\sigma}(l(x_0))$ is the estimated variance of the robust LOESS fitting in x_0 and $z_{\alpha/2}$ is the percentile of the normal distribution of order $1 - \alpha/2$, in particular we take $\alpha = 5\%$ to achieve a 95% confidence interval. Moreover, we also show on the plot the original points from which the smoothed line has been computed. In this way the users can have an immediate view of the smoothed line, which allows an easy understanding of the general effect of the feature on that sample, but then they can also understand more precisely the effective behaviour of the model by looking at the original points and the confidence region. An

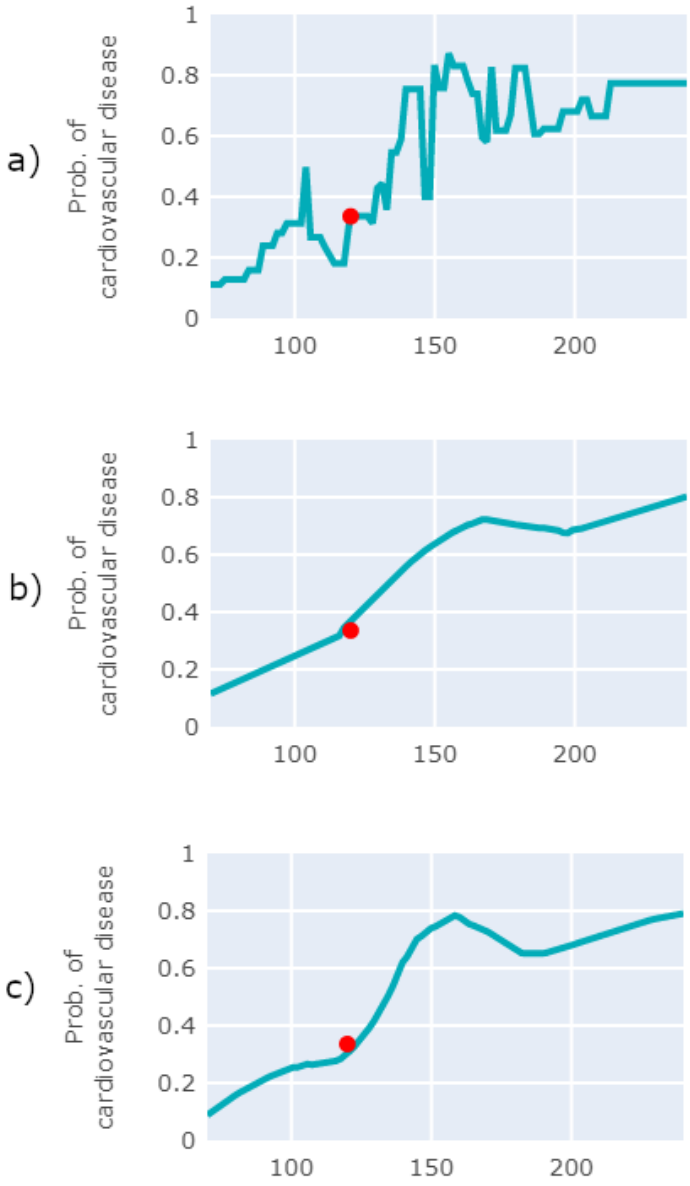

Figure 5.12: Comparison of different versions of the ceteris paribus plot for BP High feature of a sample of the cardiovascular disease dataset. In a) the original version, in b) the smoothed version with degree=1, in c) the smoothed version with degree=2

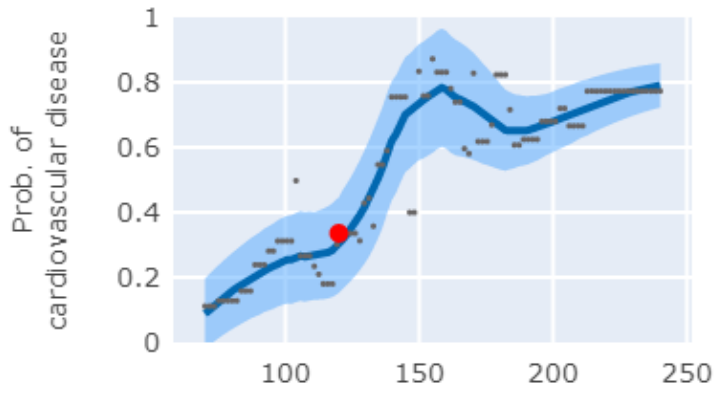

Figure 5.13: Ceteris paribus plot for the BP High of a sample of the cardiovascular disease dataset, complete with the smoothed line (dark blue), the confidence region (light blue) and the original points (grey dots). The red dot highlights the current value.

example of a ceteris paribus plot for a numerical feature, with smoothed line, confidence region and original points is in Figure 5.13.

5.4.4 Counterfactuals

Below the table with the data of the current sample there is the *VIEW COUNTERFACTUALS* button which shows to the user a pop-up containing a table where the current sample is compared with three counterfactuals. The different values are highlighted and the confidence of the prediction is reported for the counterfactuals. The three counterfactuals are computed with the three distance measures previously mentioned (Manhattan, Euclidean and Chebyschev), but the order in which they are shown is random. This view offers the possibility to understand which minimal changes could alter the prediction and so which values are more critical for the current output. The counterfactual table is shown in Figure 5.14.

5.4.5 Textual explanations & user evaluation

We show to the users two textual explanations, the first one is generated automatically using the grammar we defined, while the second one is produced by our generative language model. The users are not aware of the different

Counterfactual examples

Counterfactuals are examples from the dataset that are similar to the original sample but are classified differently. They can be useful to understand the reasons behind the classification, since they highlight minimum changes needed to switch the prediction.

Description	Age	Gender	BP high	BP low	Cholesterol	Glucose	Smoking	Alcohol	Physical Activity	BMI	Prediction	Confidence
Original Sample	50	Male	120	80	Above Normal	Above Normal	No	No	Yes	26.2	NO DISEASE	75 %
#1	***55***	Male	120	80	Above Normal	Above Normal	No	No	Yes	***24.0***	**CARDIOVASCULAR DISEASE**	52 %
#2	***51***	Male	***130***	80	Above Normal	Above Normal	No	No	Yes	***27.5***	**CARDIOVASCULAR DISEASE**	53 %
#3	***51***	Male	120	80	Above Normal	***Normal***	***Yes***	***Yes***	Yes	***27.2***	**CARDIOVASCULAR DISEASE**	59 %

Close

Figure 5.14: Counterfactual view for a sample of the cardiovascular disease dataset

sources of the two explanations. We ask the users to compare them, rating both of them with a value in [1-5] for each of the following properties:

1. clarity
2. completeness
3. correctness

Then we ask them to indicate which one they prefer. This is explicitly asked because, despite the evaluation of the above properties, there may be other factors which make them prefer one instead of the other. Then we ask them if they want to edit the second explanation. This is not mandatory, but we encourage them to do so, since it could be useful for us to see which kind of modifications the users would like to do. An example of this form is shown in Figure 5.15.

5.4.6 Q&A form

We propose the Q&A system to the users showing the form of Figure 5.16, where they can evaluate the answer for correctness, completeness and clarity and they can possibly edit it. It is not always shown, but only on the third, fifth, seventh and ninth samples and then, after the tenth sample, randomly with a probability of 20%.

5.4.7 Interface evaluation form

After the tenth sample we show an additional form with some questions to evaluate the interface, the system in general and to have a general comparison of the two types of explanations (baseline vs gpt2) on the two datasets. This form is shown in Figure 5.17.

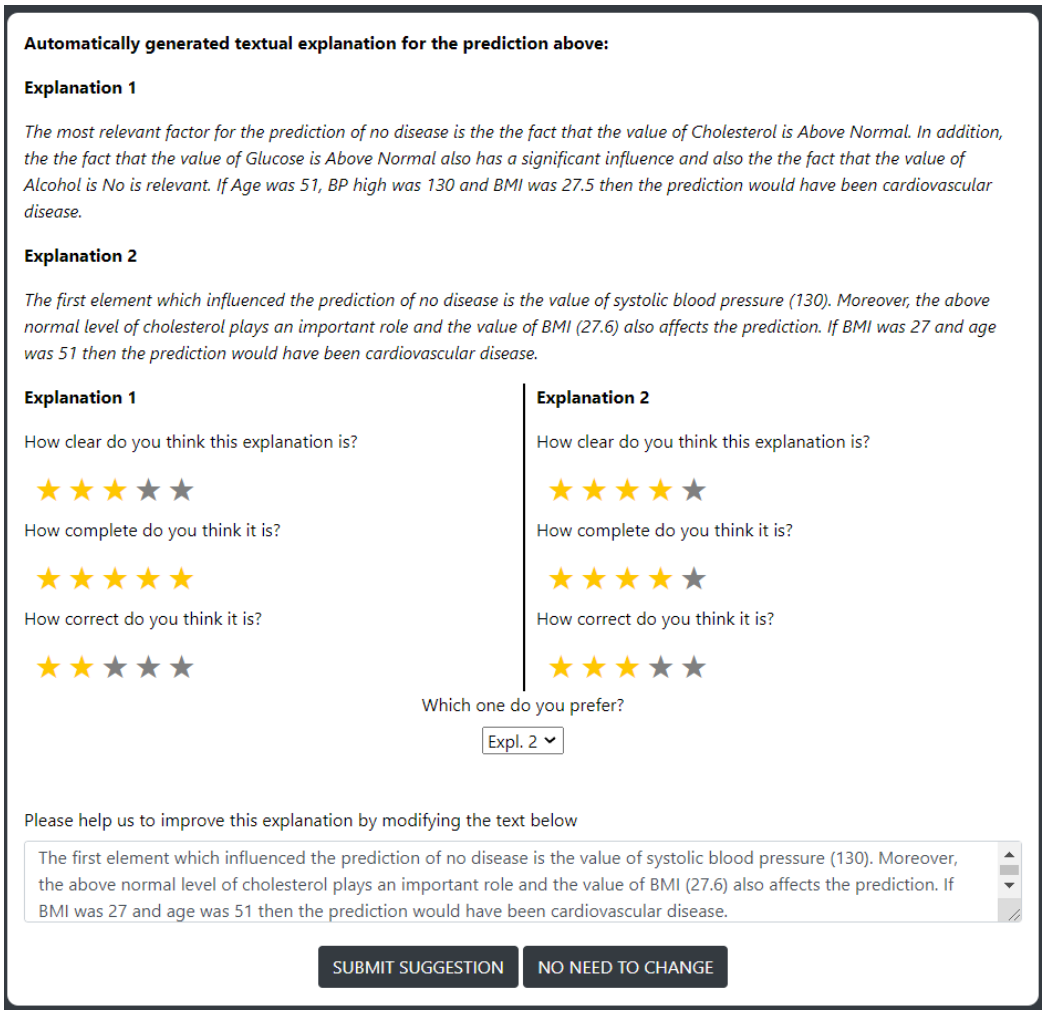

Figure 5.15: The form with the textual explanations and the user evaluation

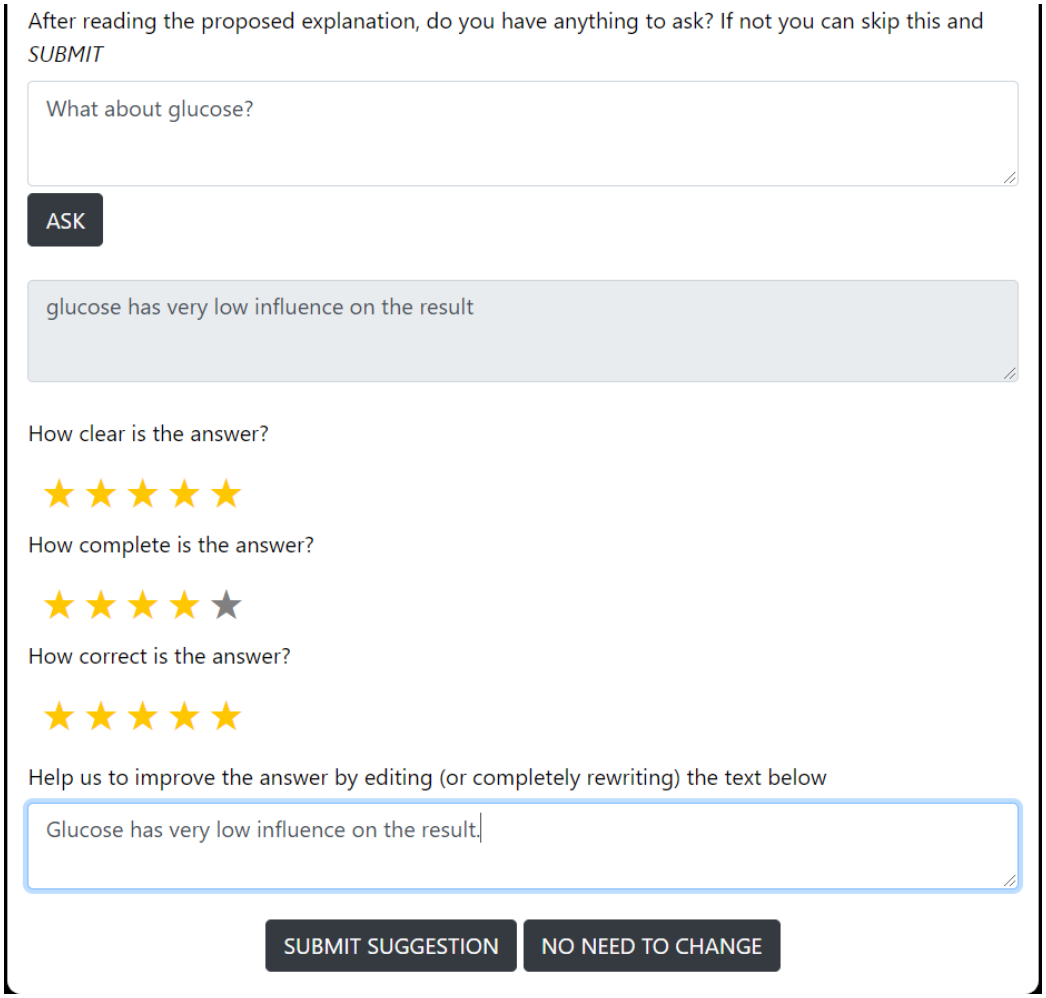

Figure 5.16: Q&A form

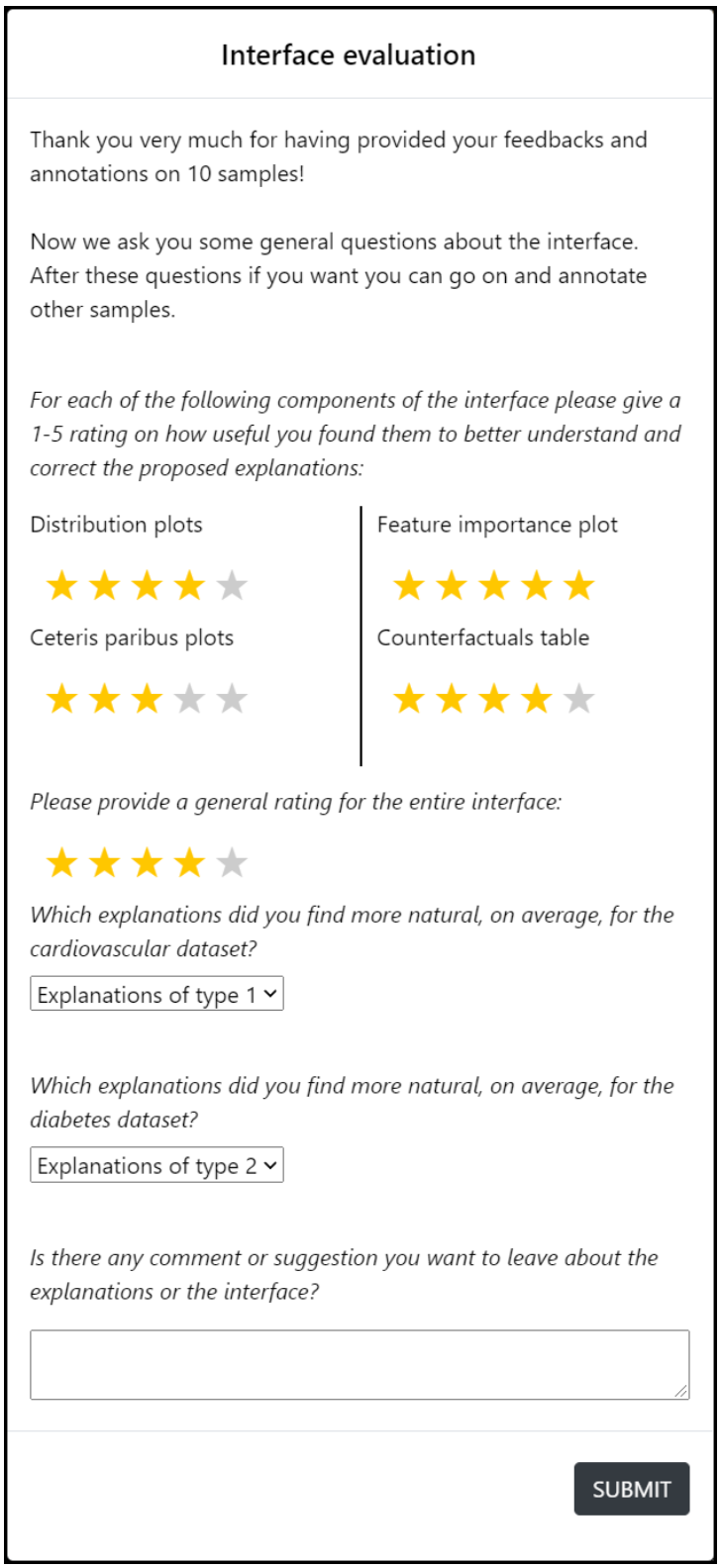

Figure 5.17: Final evaluation form

Chapter 6

Experiments and evaluation

In this chapter we present the results of our experiments, including the user-study. In Section 6.1 we report the results of the different classifiers we have considered on the two classification datasets (the models we want to explain). In Section 6.2 we briefly detail the training parameters of our language models, while in Section 6.3 we discuss how we evaluate them. In Section 6.4 we present the results of the experiments and of the user-study for the explanations model and in Section 6.5 we present the same results for the Q&A model. Finally, in Section 6.6 we present the results related to the evaluation of the interface.

6.1 Classifiers

The results of the various classifiers that we use as black-box models to be explained are measured in terms of accuracy with 10-fold cross-validation. They are reported in Table 6.1 for the cardio dataset and in Table 6.2 for the diabetes dataset. In both cases the results are in line with those reported by Kaggle users on the datasets pages. More details on the training are in Appendix A.

6.2 Common aspects of GPT-2 trainings

We use the HuggingFace's *Transformers* library to train all the GPT-2 models in our experiments. The optimizer is AdamW [54], with default parame-

Model	CV Accuracy
XGBoost	0.752
Random Forest	0.688
Logistic Regression	0.728
Neural network	0.735

Table 6.1: CV results for different classifiers on the cardiovascular disease dataset

Model	CV Accuracy
XGBoost	0.750
Random Forest	0.766
Logistic Regression	0.770
Neural network	0.763

Table 6.2: CV results for different classifiers on the pima diabetes dataset

ters $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, $lr = 5 \cdot 10^{-5}$ and a linear scheduler. These values are suggested by HuggingFace as optimal[1]. We use the small version of GPT-2, due to hardware limitations and considering that our dataset is not particularly large. We split the dataset between training and validation as 90%/10%.

We initially pre-train our language models on the MedDialog-EN dataset, so to give them a basic knowledge of the medical domain. We train them until the validation perplexity stops improving (Figure 6.1).

6.3 Evaluation of explanations and Q&A

In this section we present the methods that we use to evaluate our models. In Section 6.3.1 we discuss the different automatic metrics that we have considered, while in Section 6.3.2 we motivate the use of a user-study for the final evaluation of the system.

[1] `https://huggingface.co/transformers/main_classes/trainer.html#transformers.TrainingArguments`

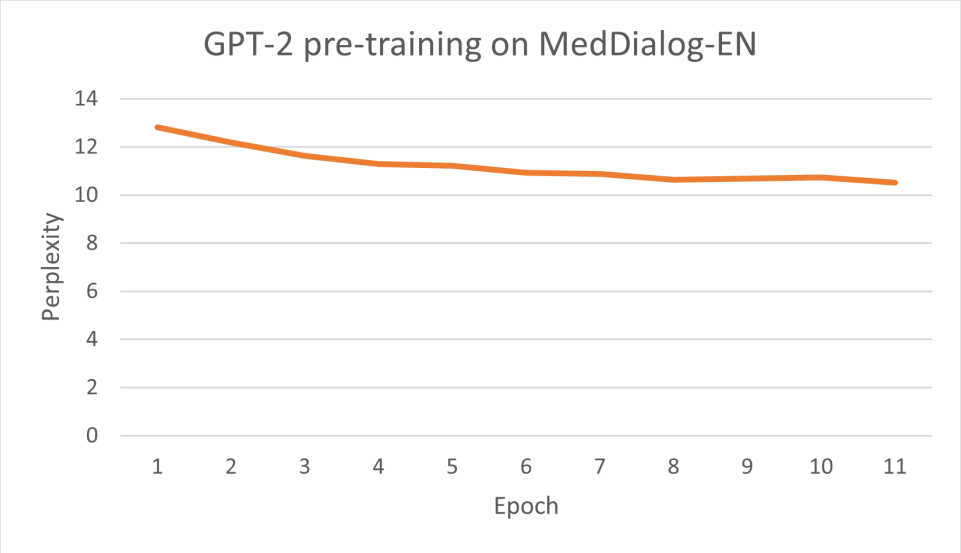

Figure 6.1: GPT-2 Pre-training on MedDialog-EN dataset

6.3.1 Automatic metrics

For both the textual explanations and the Q&A system we consider some metrics that we can compute on the validation set to understand the performances of different versions of our model, different input encodings and different output generation methods. In particular we consider the cumulative BLEU score [55], the METEOR score [56] and the BLEURT score [57].

The BLEU score was proposed for machine translation, but it can be used for many NLP tasks, wherever there is a comparison between a reference and a candidate. The BLEU score in its basic form is a modified precision over the n-gram matching between a candidate sentence and one or more reference sentences. Its cumulative version is the geometric mean of the basic n-gram versions with n from 1 to N, where N is typically 4, multiplied by a brevity penalty.

The METEOR score instead computes the best word-alignment between the candidate and a reference, considering also stemming and synonyms from the WordNet dictionary, and then it measures a parametric armonic mean of precision and recall on this alignment. Finally this result is multiplied by a penalty relative to the order of the words.

The BLEURT is representative of a different class of methods: embedding-based models, trained to evaluate text. They use BERT, with a training set

composed of tuples of the type (reference, candidate, human score) and a linear layer over BERT so to make it predict the score. Since the human-annotated dataset is very small, a key point is a pre-training over a large dataset of pairs of (sentence, automatically perturbed sentence), where the reference score is also automatically calculated on the base of the type of perturbation.

We consider BLEU with N=4, since it is the most used metric in NLP, and METEOR, since it takes into account synonyms, which can be particularly relevant for our task, and it has been seen to have a better correlation with human judgement with respect to BLEU. We take into account BLEURT so to have a different type of metric and also because there is an analysis on a (small) dataset of explanations which highlights a better correlation of BLEURT with human judgement with respect to BLEU, METEOR and other metrics [58].

We can observe with an example how the different metrics behave with two different sentences (reference and candidate) which have the same meaning, but expressed using different words:

- *The glucose is also an important element for the result.*
- *In addition, glucose has a significant effect on the diagnosis.*

The results are the following: BLEU-4 = 0, METEOR=0.150, BLEURT=0.674 . It is clear that BLEURT has a better ability of recognizing the meaning behind the sentences, and this is also partially present in METEOR, while BLEU is really focused on the words. Nevertheless, we will see a general agreement between the three metrics during the comparisons of different versions/encodings, not in the absolute values but in the rankings they determine.

We use the same metrics for the the Q&A system, with the only difference that we use the cumulative BLEU-1 instead of the cumulative BLEU-4, since the answers of the second type are very short and they typically do not reach a 4-gram, making impossible to use BLEU-4. We use BLEU-1 since it is more meaningful to assess the single words which compose the encoded answer in this case.

When we have to perform significance tests to assess the difference between two models/versions, we use the approximate randomization technique

presented in [59], where we shuffle the test output of the two models, we measure the difference in the metric values and we compute the p-value as the the fraction of iterations in which the difference results to be higher than the one originally measured between the two versions. This type of test avoids to make the independence assumption typical of many statistical tests, which is often violated in many NLP tasks [60], like ours, where we compare similar models or identical models with different input formats.

6.3.2 User study

None of these automatic metrics is perfect and considering that our purpose is to develop textual explanations that aim to be particularly useful for end-users, we include also a user-study to evaluate the performances. Another important reason for the user study is that we can use all these automatic metrics only in a comparison between our generative model output and the baseline grammar output, but this is not a very good way of judging it, since we would like to obtain better performances than the baseline grammar and this is hard to capture by automatic metrics which take the grammar output as a reference, even if some of them should be able to understand a certain level of variability which keeps the same meaning of the reference sentence. Instead, users can express their evaluations and compare the results according to their opinions, which is what we are truly interested in. At the same time we cannot avoid to use automatic metrics because of the difficulties related to a user study, that we can perform only at the end of our development process. Our user study exploits the interface presented in Section 5.4 and the details on its results are presented in the next sections.

6.4 Textual Explanations

In this section we present the results for the textual explanations, considering different input encodings, explanation versions and augmentation levels. We use the automatic metrics discussed in the previous section on the cardiovascular disease dataset, we compare these results with the ones on diabetes dataset and we present the results of the user study on the final version of the system, for both datasets.

6.4.1 Experiments

In this section we refer to the various encodings that we have presented in Section 5.2.3. We briefly summarize their characteristics:

- **Encoding 0**: all feature values with the names of the three most relevant features at the end
- **Encoding 1**: similar to *Encoding 0*, but it separates more clearly the features values and it reports also the Shapley values of the three most important features
- **Encoding 2**: adds the Shapley values to every feature
- **Encoding 3**: *Encoding 2* without the list of the three most important features
- **Encoding 3.1**: includes mean and standard deviation for each numerical feature
- **Encoding 3.2**: includes an indication of the position of the value with respect to the mean (higher/lower/similar) and the number of standard deviations it is above/below the mean, for each numerical feature
- **Encoding 3.1.1**: includes 4 y-values of the ceteris paribus plot, for each numerical feature
- **Encoding 3.1.2**: includes 2 words to describe the high and low values of the ceteris paribus plot (which can be high, low or without a clear behaviour), for each numerical feature

Version 1: Feature importance

In the first version of the textual explanations we focus on communicating to the user the most relevant features which determine the output. We start to consider *Encoding 0* and we measure BLEU, METEOR and BLEURT to determine the number of epochs for the training. The results are in Table 6.3. Since all the metrics are reducing we decide to stop at the first epoch.

These results have been obtained with the GPT-2 generation parameters listed in Table 6.4. We run a grid search over the various GPT-2 generation

N. Epochs	BLEU	METEOR	BLEURT
1	0,408	0,587	0.680
2	0,401	0,494	0.673
3	0,395	0,477	0.671

Table 6.3: Performances on validation set with encoding 0

Repetition Penalty	1,5
No Repeated n-gram size	2
Top k	125
Top p	0.92
Temperature	0.85

Table 6.4: GPT-2 generation parameters used during the first test with encoding 0

hyperparameters to determine the best values, with the ranges reported in Table 6.5, in addition to beam and greedy search.

Repetition Penalty	[1, 1.5]
No Repeated n-gram size	[0, 1, 2]
Top k	[10, 50, 100, 1000]
Top p	[0.90, 0.93, 0.96, 0.99]
Temperature	[0.7, 0.8, 0.9]

Table 6.5: Ranges for GPT-2 parameters grid search

We report all the results in Appendix B. There are two combinations of parameters that are the best for both BLEU and METEOR and near the best for BLEURT. We pick the one which makes the generation procedure simpler: repetition penalty = 1, no repeated n-gram size = 0, top-k = 10, top-p = 0.9, temperature=0.7.

Considering these generation parameters, we now compare the four different encodings proposed in Section 5.2.3. The results are in Table 6.6. As we can see there is no difference between *Encoding 2* and *3*, while their results are clearly better than *Encoding 0* and *Encoding 1*. We decide to use *Encoding 3*, since in absence of differences we prefer the simplest one.

We consider also a variation of *Encoding 3* where we rename the features with more appropriate names (Table 6.7). This new version improves

Encoding	BLEU	METEOR	BLEURT
0	0,443	0,621	0.680
1	0,445	0,628	0.691
2	**0,470**	**0,658**	**0.718**
3	**0,469**	**0,661**	**0.718**

Table 6.6: Comparison between different encodings on version 1 of the explanations

Original Name	New Name
ap_lo	diastolic blood pressure
ap_hi	systolic blood pressure
gluc	glucose
active	physical activity
alco	alcohol
smoke	smoking

Table 6.7: Features renaming

significantly both BLEU and METEOR, which reach **0.582** and **0.722**, respectively, and also BLEURT is increased to **0.730**. Considering this, we keep the renaming of the features for all the following experiments.

To increase the variability of the training set we apply the augmentation based on BERT, randomly masking a word from the explanation and replacing it with the one predicted by BERT. We avoid to mask the feature values, which could not be predicted by BERT, and also the feature names, whose change may confuse the users. We compare the results obtained by applying this technique one, two or three times in Table 6.8.

Augmentation	BLEU	METEOR	BLEURT
No augmentation	0.582	0.722	0.730
BERTx1	0.571	0.716	0.718
BERTx2	0.554	0.705	0.704
BERTx3	0.536	0.696	0.695

Table 6.8: Comparison of different augmentations on version 1 with encoding 3

In Table 6.9 we show the effect on the dataset of the various augmentation levels, in terms of number of changed samples and in term of average Lev-

enshtein distance, a measure of distance between strings, corresponding to the minimum number of characters which need to be changed to move from one to the other, considering insertion, deletion and substitution as atomic operations [61].

Augmentation	% of different samples	Avg Lev. Dist.
Base vs BERTx1	53%	3.92
Base vs BERTx2	76%	7.67
Base vs BERTx3	88%	11.29
BERTx1 vs BERTx2	51%	3.76
BERTx2 vs BERTx3	49%	3.65

Table 6.9: Effect of augmentation steps on the dataset

The augmentation seem to reduce the performances, but this is not surprising, since we are introducing more variability in the dataset and it is not necessarily negative. The difference between no augmentation and BERTx1 is significant at 5% for BLEU and METEOR, but not for BLEURT, while the difference between BERTx1 and BERTx2 is significant for all. Considering that the augmentation sometimes can damage the explanations, introducing some words which are not very appropriate (eg: *noxious* disease instead of *cardiovascular* disease), and that the effect of the various augmentation levels in term of number of changed samples progressively reduces, we decide to limit the augmentation at BERTx1.

Version 2: Including Statistics Information

In the second version of our explanations we include an additional description related to the values of numerical features. In particular, we would like to make our system able to tell when a value is particularly high or low with respect to the mean, in term of number of standard deviations. This is basically a description of the information encoded by the distribution plots of our interface (see 5.4.2). We compare the two encodings proposed in Section 5.2.3, applying the BERT augmentation, and we report the results in Table 6.10. Considering the similarity of the results, we decide to keep *Encoding 3.1* which is simpler.

Encoding	BLEU	METEOR	BLEURT
3.1	0.604	0.732	0.721
3.2	0.606	0.735	0.732

Table 6.10: Comparison of two encodings for version 2

BLEU	METEOR	BLEURT
0.541	0.713	0.693

Table 6.11: Results on version 3

Version 3: Including Counterfactual

In the third version of our explanations we also include the description of a counterfactual, highlighting what small changes could be made to alter the result. This requires to encode in the input the counterfactual, in particular we use the Manhattan counterfactual and we encode only the features which present a minimal change ($> 5\%$ of range for numerical features) with respect to the current sample. We encode it as proposed in Section 5.2.3. Results for this version are in Table 6.11.

Version 4: Including Ceteris Paribus Information

In the final version of the interface we would like to include also a motivation for the effect of a certain feature, exploiting the information encoded in the *ceteris paribus* plot. It is often the case that the plot allows to say that high/low values of a feature, for the current sample, are associated with a high/low probability of disease. We try to add this to our explanations. Moreover, we also extend the grammar to introduce more variability.

The comparison of the two variants, without extended grammar, is in Table 6.12, while the results with the extended grammar are in Table 6.13. Both with extended and non-extended grammar the difference between the two encodings is not statistically significant at 5%, we decide to use the second one.

The extended grammar reduces the values of metrics, but we decide to keep it in the final version because it gives more variability to the text and this reduction is partially due to the various forms in which the same explanations

Encoding	BLEU	METEOR	BLEURT
v4.1	0.590	0.756	0.670
v4.2	0.587	0.754	0.669

Table 6.12: Comparison of two encodings for version 4

Encoding	BLEU	METEOR	BLEURT
v4.1	0.519	0.705	0.633
v4.2	0.527	0.713	0.640

Table 6.13: Comparison of two encodings for version 4 with extended grammar

can be expressed, which may not always matched by the automatic metrics, even when their meaning is exactly the same.

Results on diabetes dataset

We compare now the results on the pima diabetes dataset, considering the same model used so far, trained only on the explanations for the cardiovascular disease dataset, and the same model fine-tuned on explanations for the diabetes dataset, generated with the same rules. We summarize the results in Table 6.14.

We can observe the big difference which is present between the two models. The base model, trained only on cardio explanations, exhibits poor performances on this dataset and it often speaks about features which do not exist in it, trying to find reference to its known features of the cardio-

Version	Base Model			FT Model		
	BLEU	METEOR	BLEURT	BLEU	METEOR	BLEURT
1	0.197	0.384	0.410	0.430	0.608	0.567
2	0.233	0.378	0.420	0.484	0.644	0.568
3	0.182	0.390	0.413	0.408	0.611	0.578
4	0.282	0.460	0.377	0.495	0.768	0.566
4*	0.239	0.413	0.393	0.584	0.688	0.522

Table 6.14: Results on the diabetes dataset for the base model (trained only on cardio explanations) and the fine-tuned one. Version 4 is the v4 with extended grammar*

vascular disease dataset. Instead with fine-tuning, even if the fine-tuning dataset is very small (hundreds of samples vs tens of thousands of the cardio dataset), the results are quite good and not far from the ones achieved by the original model on the cardio dataset. This highlights a limitation of the model, but also its ability to perform well when it is fine-tuned. We report below an example of explanations generated by the original model and by the fine-tuned model, to give an idea of the big difference that can be present:

- **Original model:** *The most relevant factor for the prediction of no disease is the fact that the patient is young, where low values of this feature are associated with a low probability of cardiovascular disease. The value of systolic blood pressure (120) is also an important element, where low values of this feature are associated with a low probability of cardiovascular disease. The normal level of cholesterol is also a cause. If BMI was 23 and age was 39 then the classifier would have predicted cardiovascular disease.*

- **Fine-tuned model:** *The main reason why he has been predicted as having absence of disease is the value of BMI (26), which is lower than the mean, where low values of this feature are associated with a low probability of diabetes. The value of glucose (111), which is lower than the mean, plays an important role, where low values of this feature are associated with a low probability of diabetes, and the value of age (24), which is lower than the mean, also contributes to the result, where low values of this feature are associated with a low probability of diabetes. If age was 34.0, insulin was 79.0, glucose was 130.0, skin thickness was 23.0, blood pressure was 78.0 and diabetes pedigree function was 0.325 the opposite would have been predicted.*

In this example the first one is completely wrong, while the second one is correct.

6.4.2 User study results

We have collected a total of 235 annotations on explanations, coming from 34 users. Not all users have completed the 10 requested samples, we have 145 annotations on samples from the cardiovascular disease dataset and 90

EXPL TYPE	DATASET	1	2	3	4	5	AVG
GRAMMAR	Cardio	1%	5%	16%	50%	28%	3.98
GPT-2	Cardio	3%	5%	19%	27%	47%	**4.10**
GRAMMAR	Diabetes	0%	2%	20%	47%	31%	**4.07**
GPT2 - TL	Diabetes	21%	11%	18%	16%	34%	3.32
GPT2 - FT	Diabetes	6%	15%	25%	25%	29%	3.56

Table 6.15: Clarity rates for textual explanations

EXPL TYPE	DATASET	1	2	3	4	5	AVG
GRAMMAR	Cardio	1%	8%	12%	30%	50%	**4.20**
GPT2	Cardio	1%	5%	23%	34%	37%	4.01
GRAMMAR	Diabetes	0%	7%	16%	37%	41%	**4.12**
GPT2 - TL	Diabetes	42%	8%	26%	18%	5%	2.37
GPT2 - FT	Diabetes	8%	12%	27%	29%	25%	3.52

Table 6.16: Completeness rates for textual explanations

over the pima diabetes dataset, of which 52 using the fine-tuned model and 38 with the model trained only on cardio explanations.

For every sample we have asked the users to evaluate clarity, completeness and correctness of both the grammar based and the gpt-2 based explanation. We report the results in Tables 6.15, 6.16 and 6.17 and we show them with plots in Figures 6.2, 6.3, 6.4, 6.5, 6.6 and 6.7, with distinction on the diabetes dataset between *GPT2 - FT* for the fine tuned model and *GPT2 - TL* for the model trained only on cardio dataset.

We can observe that on the cardio dataset the gpt-2 based explanations result slightly more clear and slightly less complete than the grammar-based

EXPL TYPE	DATASET	1	2	3	4	5	AVG
GRAMMAR	Cardio	1%	7%	14%	35%	42%	**4.10**
GPT2	Cardio	5%	8%	28%	28%	31%	3.73
GRAMMAR	Diabetes	1%	4%	18%	38%	39%	**4.09**
GPT2 - TL	Diabetes	61%	8%	13%	11%	8%	1.97
GPT2 - FT	Diabetes	13%	17%	17%	35%	17%	3.25

Table 6.17: Correctness rates for textual explanations

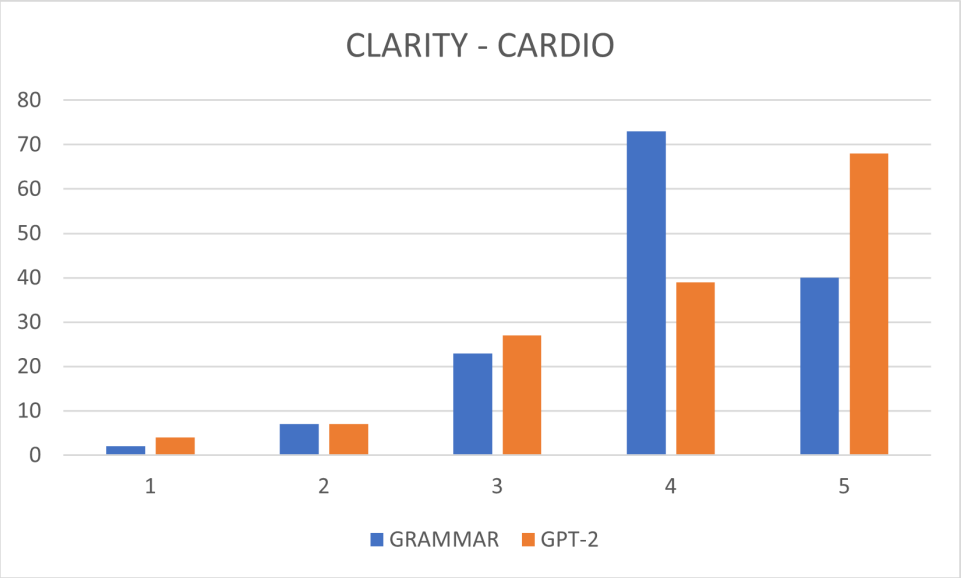

Figure 6.2: Clarity rates on explanations for cardio dataset

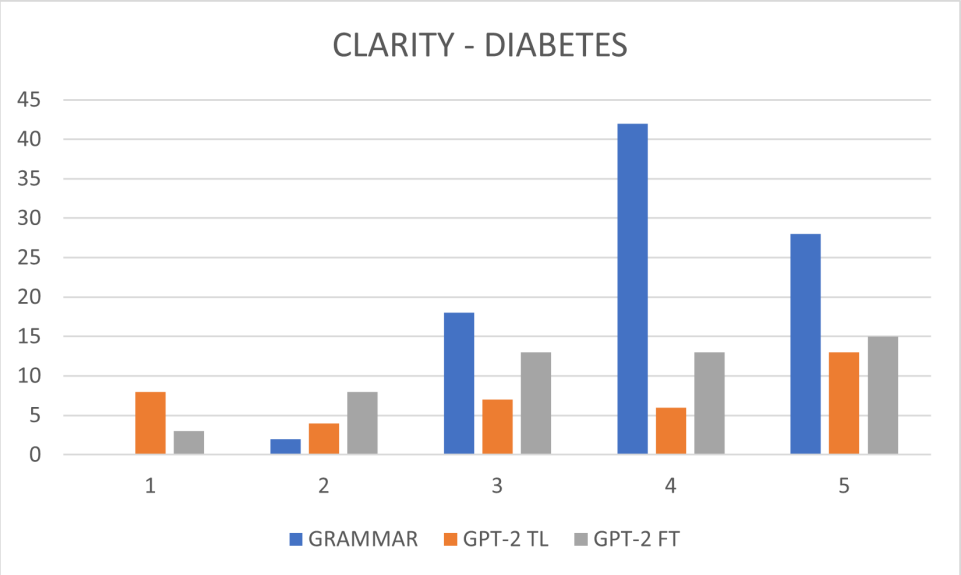

Figure 6.3: Clarity rates on explanations for diabetes dataset

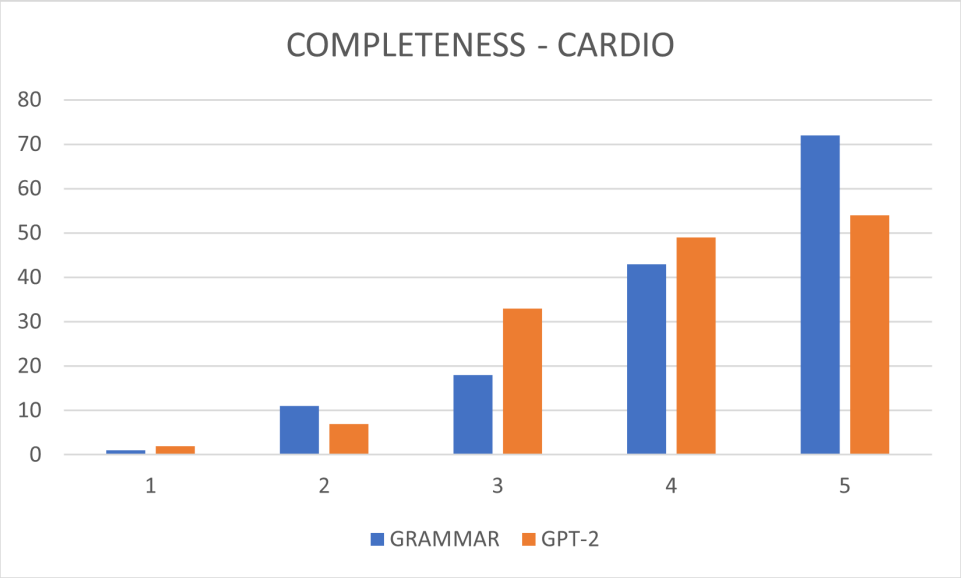

Figure 6.4: Completeness rates on explanations for cardio dataset

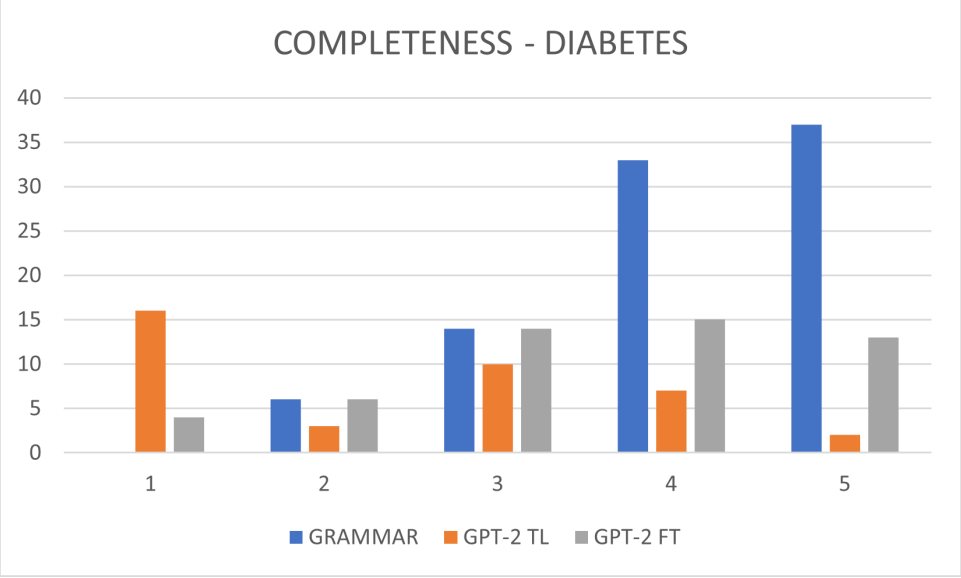

Figure 6.5: Completeness rates on explanations for diabetes dataset

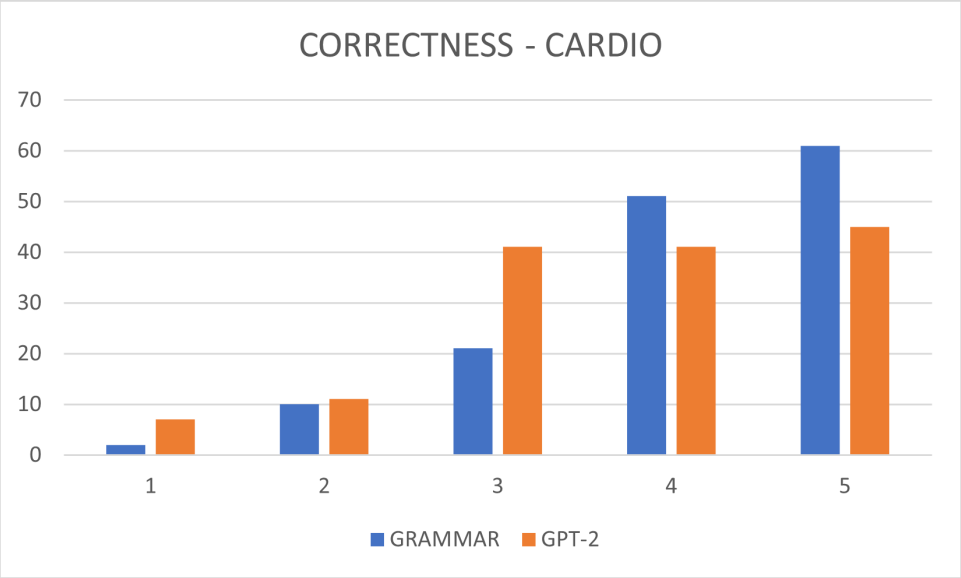

Figure 6.6: Correctness rates on explanations for cardio dataset

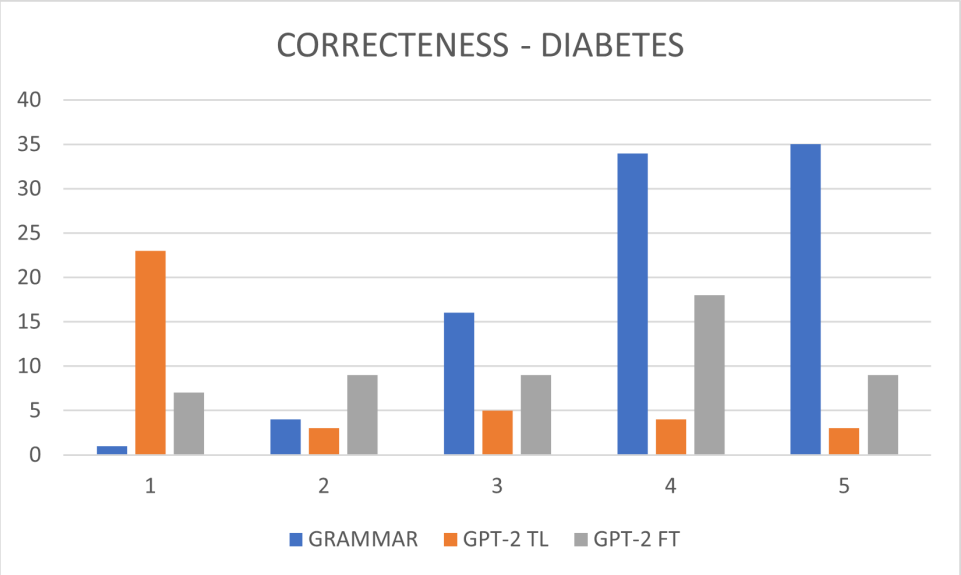

Figure 6.7: Correctness rates on explanations for diabetes dataset

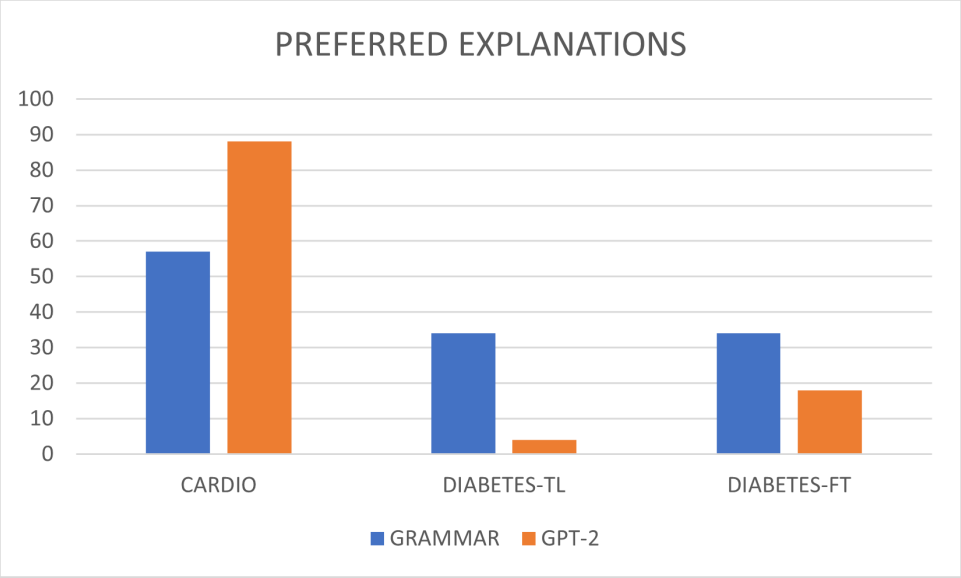

Figure 6.8: Preferred explanations (Which explanation do you prefer?)

explanations. For the correctness the grammar-based have a higher score, but the result of the gpt-2 based is near. On the diabetes dataset the transfer learning approach, using the model trained only on cardio, has poor performances, in particular for correctness. The completeness and clarity are reduced with respect to the previous results, but we think this is more a consequence of the incorrectness. The fine-tuned model achieves better results, it is above 3 in all the three scores and even above 3.5 in two of them, but it is still lower than the grammar-based.

We analyze then the answers to the question *Which explanation do you prefer?*, whose results are in Figure 6.8. For the cardiovascular disease dataset the users seem to prefer more the gpt-2 based explanation, while for the pima diabetes dataset they definitely prefer the grammar-based if it is compared with the one produced by the original model, while they still prefer it, but with a smaller difference, when compared to the one produced by the fine-tuned model. This is in line with what we expected, we were more uncertain only about the results to expect from the fine-tuned model on the diabetes dataset, since it is not bad but not as good as the original model is on the cardio dataset.

We can observe also the answers to the questions *Which explanations did you find more natural , on the average, on the cardio/diabetes dataset?*,

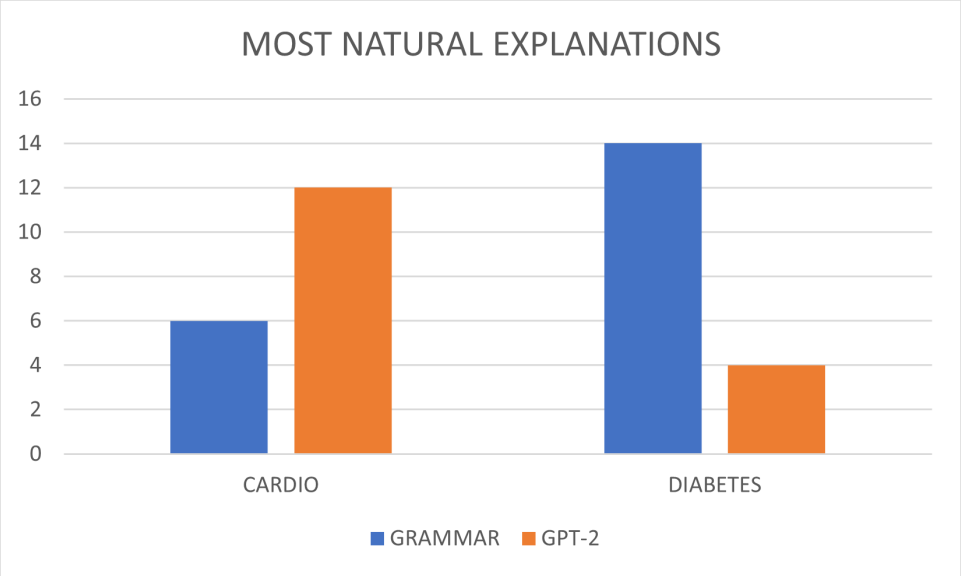

Figure 6.9: Most natural type of explanation on the two datasets according to the users who completed 10 samples

which are plotted in Figure 6.9. Here the number of answers is lower, 18, since this is the number of users who completed at least 10 samples, which is the moment when these two questions are asked.

The users were also asked to edit the second explanation, if they wanted to improve or correct it. We have recorded 70 editings, 41 of which are on the cardiovascular disease dataset, 9 are on the pima diabetes with the cardio model and the remaining 20 are on the pima diabetes with the fine-tuned model. 7 editings are not presenting any difference (eg: just a space at the end). For the cardio dataset we have:

- 15 minor changes (punctuation, substitution of a word)
- 6 feature deletions (the users wanted to remove one of the features which are mentioned in the explanation, not necessarily wrong features)
- 5 feature changes (all verified as correct)
- 2 feature additions (the users added additional fetures in the explanation)
- 5 corrections on the counterfactuals (4 of which are correct)

- 1 change of the classifier output (correct)
- 2 values corrections (both correct)
- 1 counterfactual removal
- 1 addition of explanations for the counterfactual
- 3 comments which are not really an explanation editing

For the diabetes dataset with transfer learning we have:

- 1 minor change
- 6 comments on the fact that the explanation is completely wrong
- 2 rewritings of wrong explanations

For the diabetes dataset with fine-tuning we have:

- 5 minor changes
- 2 feature deletions
- 3 feature changes (all verified as correct)
- 2 feature additions
- 1 change of the classifier output (correct)
- 2 values corrections (both correct)
- 2 changes of features order (both correct)
- 1 comment which is not really an explanation editing

Many of this editings are interesting and highlight the types of errors that sometimes are made by the models.

We report an example of grammar-based, gpt2-based and user-edited explanations:

- **Grammar:** *The first element which influenced the prediction of cardiovascular disease is the value of BP high (140), which is 1 standard deviation above the mean, where high values of this feature are associated with a high probability of cardiovascular disease. In addition, the value of BMI (37), which is 2 standard deviations above the mean, also has a significant influence, where high values of this feature are associated with a high probability of cardiovascular disease, and also the value of Age (59), which is 1 standard deviation above the mean is relevant, where high values of this feature are associated with a high probability of cardiovascular disease. If Age was 52 and BP high was 120 the result would have been no disease.*

- **GPT-2:** *The first element which influenced the prediction of cardiovascular disease is the value of systolic blood pressure (140), which is 1 standard deviation above the mean, where high values of this feature are associated with a high probability of cardiovascular disease. Furthermore, the fact that the patient is middle-aged has a considerable effect, and the value of BMI (37.0), which is 2 standard deviations above the mean also affects the prediction, where high values of this feature are associated with increased likelihood of cardiovascular disease. If systolic blood pressure was 120, age was 52 then the classifier would have not diagnosed the disease.*

- **User:** *The heaviest element which influenced the prediction of cardiovascular disease is the value of systolic blood pressure (140), which is enough higher than the mean, where high values of this feature are associated with a high probability of cardiovascular disease.*

In this case there is not a big difference between the first two, the user preferred the second one, probably because it is a bit more fluent, but then in her/his editing s/he completely rewrote it, leaving only a very simple explanation. This is not common, but it remembers us that we should not overwhelm the users with too much information, since they may prefer a simpler explanation, even if it is not complete.

A final question asked the users to leave a feedback or a comment on the interface, the explanations or their experience with the system in general.

Only 4 users gave a meaningful answer, these are the main elements they highlighted about the explanations:

1. *Explanations of type 2 often contain the correct features on the cardio dataset, but not in the exact order of importance*
2. *Explanations of type 2 do not work well on the diabetes dataset*
3. *Sometimes the counterfactual expressed by explanations of type 2 is not exactly coherent with the ones in the VIEW COUNTERFACTUALS table*
4. *Some explanations contrast with common knowledge, should we trust them?*
5. *The classifier sometimes seems to be wrong*
6. *It seems that both explanations try to say the same thing, but if a doctor would tell it to me with the first one I would be a bit worried, while the second one calms me down more. Is this wanted?*
7. *There are counterfactuals, even in VIEW COUNTERFACTUALS table, whose prediction is not realistic. Maybe there should be a check to avoid unrealistic predictions (eg: a patient with 200 systolic blood pressure predicted as no disease).*

Some of these comments repeat points we have already mentioned, like the performances of gpt-2 based explanations on the diabetes dataset. Others are more interesting, like the one which highlights a particular counterfactual whose prediction is not realistic: this may seem out of scope, since it depends on the classifier, but it has a relationship with the way in which counterfactuals are computed. A manual inspection that we performed did not found any example of counterfactuals with unrealistic predictions, so it should be a rare case, but it could be worth to consider other counterfactuals generation methods among the one we mentioned in Section 2.3.4, in particular [34] includes a term which takes into account the class representativity of the counterfactual. It could be useful for this problem, but it should be adopted with care, because it may lead to counterfactuals which are too far from the current sample.

Augmentation	BLEU-1	METEOR	BLEURT
No	0.550	0.548	0.697
BERT	0.440	0.453	0.544

Table 6.18: Results for Q&A version 1

Particularly interesting is the comment about the trustness determined by the explanations if pronounced by a doctor, this somehow confirms that the gpt-2 based explanations sound more natural to the users.

6.5 Question Answering (Q&A)

In this section we present the results for the question answering system, considering three different versions with different types of questions. We use the automatic metrics discussed in Section 6.3.1 on the cardiovascular disease dataset, we compare these results with the ones of the diabetes dataset and we present the results of the user study on the final version of the system, for both datasets.

6.5.1 Experiments

Version 1: Feature Importance Questions

In the first version of our Q&A system we consider only questions related to the relevance of the various features for the outcome of the classifier. We also augment the Q&A pairs with BERT, the results are compared in Table 6.18.

The great majority (84%) of questions are altered by the augmentation, with an average Levenshtein distance of 6.27, and 75% of the answers are altered, with an average Levenshtein distance of 5.83. There is the usual negative effect of the augmentation, but, as for the explanations, we know this is partially due to the difficulties of automatic metrics in detecting the differences, and we want our model to be able to understand a larger set of questions (and possibly to produce more various answers), so we use the augmented version.

Augmentation	BLEU-1	METEOR	BLEURT
No	0.362	0.312	0.890
BERT	0.362	0.310	0.889

Table 6.19: Results for Q&A version 2

Augmentation	BLEU-1	METEOR	BLEURT
No	0.421	0.384	0.809
BERT	0.370	0.344	0.734

Table 6.20: Results for Q&A version 3

Version 2: What-if Questions

In the second version we focus on the what-if questions, like *What if BMI was 30?*. In this case we want the model to produce an encoded output which can be used to create the new sample to give to the classifier. We apply BERT augmentation, only on questions in this case, and we compare the results in Table 6.19.

We do not notice any relevant difference. After having verified that 68% of samples are actually changed by the augmentation procedure, with an average Levenshtein distance of 3.7, we keep the augmented version, since it allows for more variability in the questions. The absence of difference is probably due the fact that here only the questions are augmented, with a lower effect.

Version 3: Feature Importance and What-if Questions

In the third version we want to include both types of questions of versions 1 and 2, so we use their same training set generation procedures, generating two (question, answer) pairs for each sample of the dataset, the first one with the procedure of version 1, the second one with the procedure of version 2. Results are in Table 6.20

The results confirm the discrete ability of the model to tackle both types of questions, so this is the model that we use at the end.

Version	Base Model			FT Model		
	BLEU-1	METEOR	BLEURT	BLEU-1	METEOR	BLEURT
1	0.319	0.278	0.487	0.370	0.349	0.551
2	0.130	0.111	0.572	0.236	0.175	0.865
3	0.308	0.269	0.441	0.390	0.387	0.574

Table 6.21: Results on the diabetes dataset for the base model (trained only on cardio questions) and the fine-tuned one (FT).

Results on diabetes dataset

We assess the results on the pima diabetes dataset, considering the models trained only on the question-answer pairs for the cardiovascular disease dataset and the same models fine-tuned on question-answer pairs for the diabetes dataset, generated and augmented in the same way. We summarize the results in Table 6.21.

There is a clear improvement in the fine-tuned version, but it is much higher on the second version. This is probably due to the fact that the name-entity recognition task becomes harder with certain previously unseen features (eg: *diabetes pedigree function*).

6.5.2 User study results

In our user study we propose the Q&A system only 4 times on 10 samples, two over the cardio dataset and two over the diabetes dataset. We always use the same model, the one trained only on cardio dataset, since otherwise we would not have enough annotations for both models, considering that the use of this tool is optional for the users. On 87 times that it was shown, it was used 55 times. 3 of these 55 questions were never sent to the system (the user did not press the ASK button), so we have 52 evaluations for the answers. Among the 55 questions we have 31 what-if questions, 2 of which involve more than one feature, 16 questions about the importance or the effect of a certain feature on the result, 4 questions which are related to the system but out of the classes which are known to the Q&A system, 2 questions which are completely out of topic and 2 questions which are comments to the explanations and not real questions.

We report the answer evaluation for the 45 questions which belongs to the

SCORE	CLEAR	COMPLETE	CORRECT
1	0%	25%	25%
2	13%	13%	0%
3	38%	13%	38%
4	25%	25%	13%
5	25%	25%	25%
AVG	**3,63**	**3,13**	**3,13**

Table 6.22: Q&A results for feature importance questions on cardio dataset

SCORE	CLEAR	COMPLETE	CORRECT
1	31%	31%	38%
2	0%	6%	6%
3	19%	13%	6%
4	0%	0%	6%
5	50%	50%	44%
AVG	**3,38**	**3,31**	**3,13**

Table 6.23: Q&A results for what-if questions on cardio dataset

what-if (29) and to the feature importance (16) classes which have been answered. In particular 25 of these questions are for the cardiovascular disease dataset (17 what-if and 8 feature importance) and 20 are for the diabetes dataset (12 what-if and 8 feature importance). In Tables 6.22, 6.23, 6.24 and 6.25 there are the results divided by question type and dataset. In Figures 6.10, 6.11 and 6.12 they are summarized, divided by dataset.

We can observe that for the cardio dataset there are no particular differences between the two types of questions, both reach sufficient but not excellent scores. For the diabetes dataset instead we have higher scores on average, which may be surprising considering that the model has been trained only for the cardio dataset, but there is a very low score on the correctness of the feature importance questions. It is likely that the model is confused by the different features names to look for in the encoded input, being used only to the ones of the cardio dataset, and consequently it produces wrong answers. The what-if questions instead involve a somehow simpler task of name-value recognition and it seems able to perform well even on this previ-

SCORE	CLEAR	COMPLETE	CORRECT
1	0%	13%	50%
2	0%	13%	25%
3	0%	0%	13%
4	25%	25%	13%
5	75%	50%	0%
AVG	**4,75**	**3,88**	**1,88**

Table 6.24: Q&A results for feature importance questions on diabetes dataset

SCORE	CLEAR	COMPLETE	CORRECT
1	0%	0%	0%
2	0%	0%	9%
3	9%	9%	0%
4	18%	9%	27%
5	73%	82%	64%
AVG	**4,64**	**4,73**	**4,45**

Table 6.25: Q&A results for what-if questions on diabetes dataset

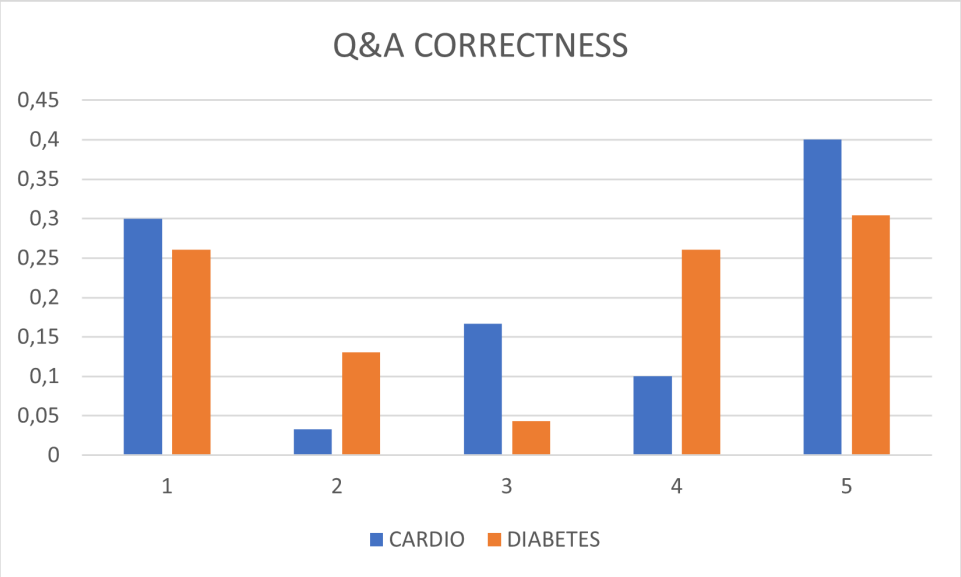

Figure 6.10: Correctness rates for Q&A

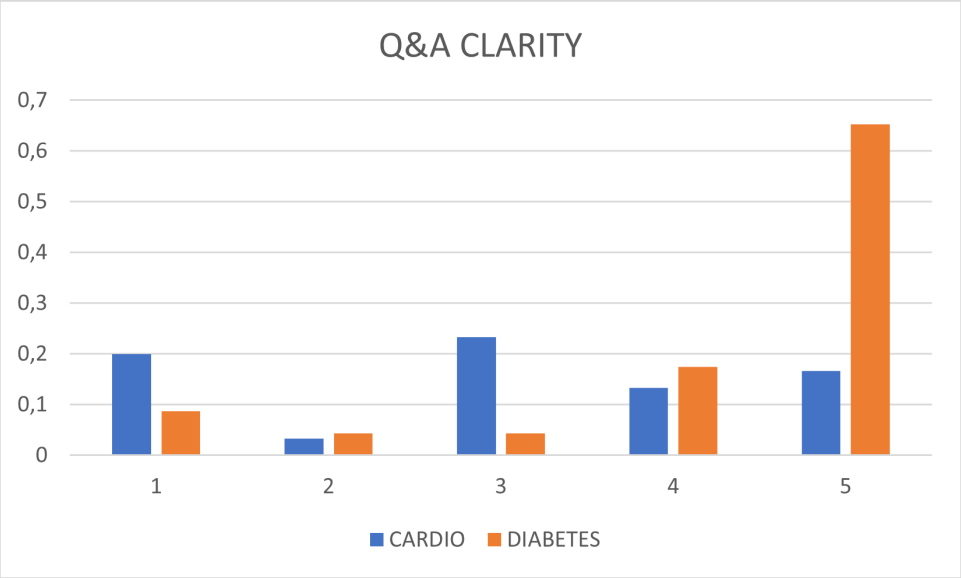

Figure 6.11: Clarity rates for Q&A

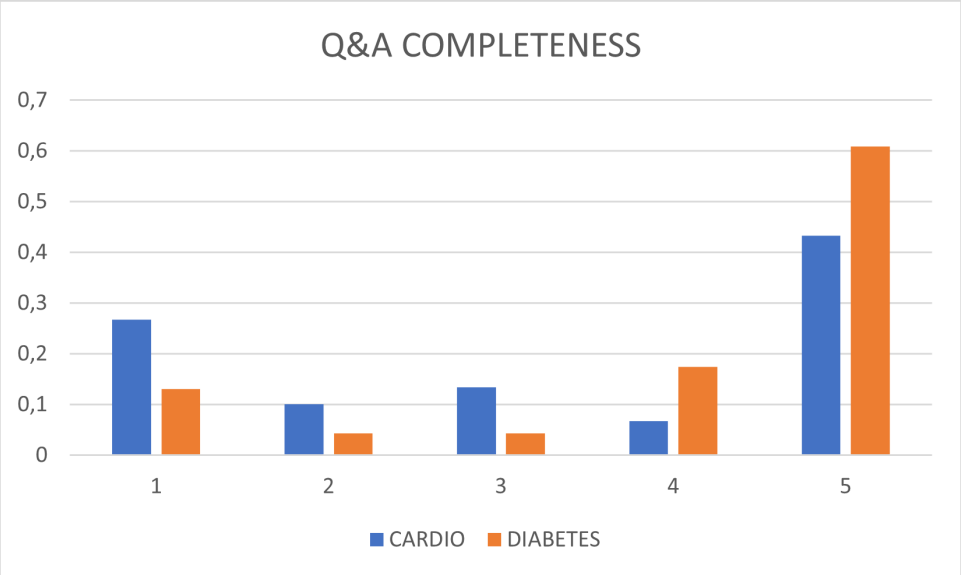

Figure 6.12: Completeness rates for Q&A

ously unseen dataset, even if this contradicts the results of automatic metrics. We trust more the users' evaluation, but we also have to consider that the number of annotations on the diabetes dataset is lower, so the results are a bit less reliable. Our manual tests make us believe that the results depend on the features: simpler features, more similar to the one already present in the cardio dataset, seem to achieve better results than the others.

The users are also asked to edit the answers if they want to correct errors or rewrite them in a better way. For the cardiovascular disease dataset we have collected 7 editings: 3 are on questions which are not of the categories known to the model, 3 are on feature importance questions and 1 is on a what-if question. For the feature importance questions we have one correction of a wrong answer, one editing with minor changes and one editing which adds the reason for the answer. The editing of the what-if answer is rephrasing the question after a wrong answer produced by the model. For the diabetes dataset we have 6 editings: 5 on feature importance questions and 1 on a what-if question. For the feature importance 3 of them correct the wrong level of importance of the feature in the model answer and 2 change the wrong feature name reported in the model answer. The editing on the what-if question removes the confidence on the result.

It is interesting to observe the questions that are not in the feature importance or what-if classes but that are still on topic:

1. *What is a healthy BMI at age 61?*
2. *What is a healthy systolic blood pressure?*
3. *What is the most common cause of cardiovascular disease?*
4. *Does he smoke?*

Apart from the last one, the other three are interesting and they should be taken into consideration in the future development of the system.

6.6 Interface

For the interface evaluation there are no automatic metrics, the only meaningful way to evaluate it is through a user-study. In this section we report the results of our-study related to the interface.

Questions/Ratings	1	2	3	4	5	AVG
DIST PLOT	0%	11%	28%	22%	39%	3.89
CP PLOT	6%	17%	17%	28%	33%	3.67
CF TABLE	11%	6%	33%	33%	17%	3.39
FEAT. IMP. PLOT	0%	0%	6%	33%	61%	4.56
INTERFACE	0%	0%	11%	33%	56%	4.44

Table 6.26: Answers to the final questions related to the interface

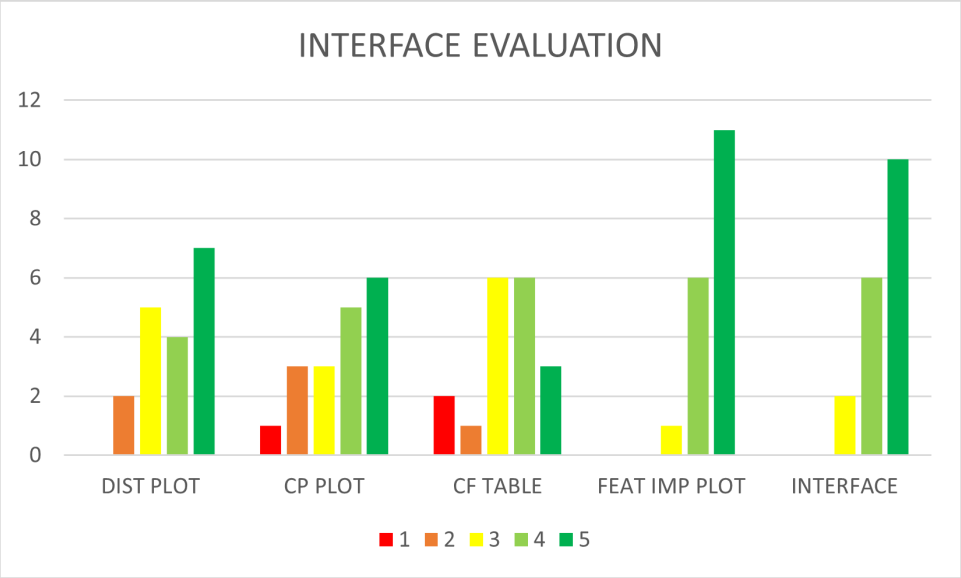

Figure 6.13: Ratings for the various elements of the interface and the entire interface

6.6.1 User study results

In the user study we present some questions regarding the interface only after the user has annotated 10 samples. These questions ask the users to provide a 1-5 rating for the various components of the interface, considering how useful and clear they found them, and a 1-5 general rating for the entire interface. Only 18 users reached this point, we report their answers about the various components and on the whole interface in Table 6.26 and in Figure 6.13.

We can observe a general positive rating for the interface. The most appreciated element is the feature importance plot, as we expected, since it is the one which is simpler to be used and that provides the most important information about feature importance. We are slighlty surprised by the lower

score achieved by the counterfactual table, it may be that the users have not fully understood the need to use the table to cross-check that portion of the textual explanations, or that these ones were in general coherent with the table in most cases and so they found the table useless.

In the final comments of the users there is only one related to the interface, which suggests to explain more in detail how to use it. We have observed some difficulties in a few users in understanding how to use the various plots, in particular the distribution plots and the ceteris paribus plots, which may need additional explanations when they are given to non-technical users. In an ideal version of the interface for real users where we have a very good textual explanation, we may even think of removing them if the users of that specific system do not find them useful.

Chapter 7

Conclusions

In this thesis we have developed a model able to produce textual explanations for a machine learning classifier and a rule-based baseline, comparing their results. We have also developed a first prototype of a question-answering system, related to the explanations, and we have embedded these models in a web interface, together with other elements which can help the users to understand the classifier behaviour and to evaluate the textual explanations. We have finally used this interface to perform a user study, with two classification datasets: the one used to build the explanation system and a different one, both in the medical domain.

In Chapter 3 we have presented our initial research questions and we can now answer them:

1. *Is it possible to produce textual explanations for a black-box classifier using a generative language model?*

   We have demonstrated that this is possible and that it can produce more natural explanations than a rule-based system, even if our model has limitations in the ability to generalize. Considering the positive effect of the fine-tuning on diabetes dataset, we believe that these limitations may be overcome if the model was trained on explanations for samples from a variety of datasets.

2. *How can we evaluate explanations in natural language?*

   We have considered different automatic metrics during the development process (BLEU, METEOR and BLEURT) and at the end we

have validated our system through a user study. Despite the effort of the NLP community in developing new metrics, more aligned with human evaluation, we believe that currently a user study is the best way of evaluating textual explanations. It is hard to cast into an automatic metric the preferences of users, which may even be quite different among them. For our development process this was particularly difficult due to the absence of a set of reference explanations evaluated as good by users. During the development, when it is not possible to perform a user study, we believe that it's better to consider different metrics, so to verify their agreement. The BLEURT metric is particularly interesting and we have observed that it seems to be able to give high scores to sentences which are quite different but with similar meaning, differently from other metrics like BLEU. Hence it should be taken in consideration, even if it is much more expensive to compute.

3. *Is it possible to allow the users to interact with a natural language explainer?*

   Our prototype of the Q&A system demonstrated that it is possible. There are still large margins of improvement, in both the correctness of the answers and the type of questions that it is able to answer, but the results are already sufficient and it suffers less of the generalization problem encountered by the explanations model.

4. *Which elements of an interface can help users to understand the behaviour of a black-box classifier?*

   According to the results of our user study, users have particularly appreciated the feature importance plot, based on Shapley values. The distribution plots and the ceteris paribus plots seem to be useful, but some users had some problems in understanding them. The counterfactual table has achieved the lower score (3.4/5), but it is still a sufficient score. We believe that it could possible to show it in different ways which may result to be simpler for the users. The general rating for out interface is pretty high (4.44) and we consider it as a useful starting point for a good xAI interface which can help end-users of a classification system.

7.1 Future perspectives

There are different aspects of the system that can be further developed. First of all it is important to extend the generalization capabilities of the explainer, so that it can manage in a better way different and previously unseen datasets. This could be achieved by training it on data coming from different datasets, but further tests need to be done to verify this.

A second point could be the extension of the explanations, making them even more varied than they currently are. This may be achieved with a dataset of human-written explanations, possibly built via crowdsourcing, exploiting the same interface we used for our user study, but asking the users to provide their own explanations, based on the other elements of the interface, without showing them our textual explanations.

A final important point is the extension of the Q&A system, which could be made able to answer more types of questions, until the ideal point of a fully conversational explainer, where we do not even start with an explanation, but we only answer to the user's questions until s/he is satisfied. This would be an ideal result, but at the moment it's not clear how it could be achieved, more investigations need to be done on this path.

Bibliography

[1] John McCarthy, Marvin L Minsky, Nathaniel Rochester, and Claude E Shannon. A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955. *AI magazine*, 27(4):12–12, 2006.

[2] Amirhosein Toosi, Andrea G Bottino, Babak Saboury, Eliot Siegel, and Arman Rahmim. A brief history of ai: how to prevent another winter (a critical review). *PET clinics*, 16(4):449–469, 2021.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[4] Spyros Makridakis. The forthcoming artificial intelligence (ai) revolution: Its impact on society and firms. *Futures*, 90:46–60, 2017.

[5] Marsden, Paul. Artificial intelligence timeline infographic – from eliza to tay and beyond, 2017. URL `https://digitalwellbeing.org/wp-content/uploads/2017/08/Artificial-Intelligence-AI-Timeline-Infographic.jpeg`.

[6] Alec Radford and Jeffrey Wu. Rewon child, david luan, dario amodei, and ilya sutskever. 2019. *Language models are unsupervised multitask learners. OpenAI Blog*, 1(8):9, 2019.

[7] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, pages 4768–4777, 2017.

[8] Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., USA, 1 edition, 1997. ISBN 0070428077.

[9] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[10] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[11] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural networks*, 3(5):551–560, 1990.

[12] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[13] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *The journal of machine learning research*, 3:1137–1155, 2003.

[14] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, 2019.

[18] Paweł Budzianowski and Ivan Vulić. Hello, it's gpt-2-how can i help you? towards the use of pretrained language models for task-oriented dialogue systems. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 15–22, 2019.

[19] Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and William B Dolan. Dialogpt: Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, 2020.

[20] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[21] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[22] Robert Neches, William R Swartout, and Johanna D Moore. Explainable (and maintainable) expert systems. In *IJCAI*, volume 85, pages 382–389. Citeseer, 1985.

[23] Michael Van Lent, William Fisher, and Michael Mancuso. An explainable artificial intelligence system for small-unit tactical behavior. In *Proceedings of the national conference on artificial intelligence*, pages 900–907. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2004.

[24] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, pages 3145–3153. PMLR, 2017.

[25] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[26] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and

saliency maps. In *In Workshop at International Conference on Learning Representations*. Citeseer, 2014.

[27] J Springenberg, Alexey Dosovitskiy, Thomas Brox, and M Riedmiller. Striving for simplicity: The all convolutional net. In *ICLR (workshop track)*, 2015.

[28] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *The Journal of Machine Learning Research*, 11: 1803–1831, 2010.

[29] Lloyd S Shapley. *Notes on the N-person Game*. Rand Corporation, 1951.

[30] Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41(3):647–665, 2014.

[31] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

[32] Shubham Rathi. Generating counterfactual and contrastive explanations using shap. *arXiv preprint arXiv:1906.09293*, 2019.

[33] Rafael Poyiadzi, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. Face: Feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 344–350, 2020.

[34] Arnaud Van Looveren and Janis Klaise. Interpretable counterfactual explanations guided by prototypes. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 650–665. Springer, 2021.

[35] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160, 2018.

[36] Mireia Ribera and Agata Lapedriza. Can we do better explanations? a proposal of user-centered explainable ai. In *IUI Workshops*, volume 2327, page 38, 2019.

[37] Q Vera Liao, Daniel Gruen, and Sarah Miller. Questioning the ai: informing design practices for explainable ai user experiences. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–15, 2020.

[38] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38, 2019.

[39] Richard Dazeley, Peter Vamplew, Cameron Foale, Charlotte Young, Sunil Aryal, and Francisco Cruz. Levels of explainable artificial intelligence for human-aligned conversational explanations. *Artificial Intelligence*, 299:103525, 2021.

[40] Thomas P Quinn, Stephan Jacobs, Manisha Senadeera, Vuong Le, and Simon Coghlan. The three ghosts of medical ai: Can the black-box present deliver? *Artificial Intelligence in Medicine*, page 102158, 2021.

[41] Rita Sevastjanova, Fabian Beck, Basil Ell, Cagatay Turkay, Rafael Henkin, Miriam Butt, Daniel A Keim, and Mennatallah El-Assady. Going beyond visualization: Verbalization as complementary medium to explain machine learning models. In *Workshop on Visualization for AI Explainability at IEEE VIS*, 2018.

[42] Jinkyu Kim, Anna Rohrbach, Trevor Darrell, John Canny, and Zeynep Akata. Textual explanations for self-driving vehicles. In *Proceedings of the European conference on computer vision (ECCV)*, pages 563–578, 2018.

[43] Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. Generating visual explanations. In *European conference on computer vision*, pages 3–19. Springer, 2016.

[44] Michał Kuźba and Przemysław Biecek. What would you ask the machine learning model? identification of user needs for model explanations based on human-model conversations. In *Joint European Conference on*

Machine Learning and Knowledge Discovery in Databases, pages 447–459. Springer, 2020.

[45] Christian Werner. Explainable ai through rule-based interactive conversation. In *EDBT/ICDT Workshops*, 2020.

[46] Q Vera Liao, Milena Pribić, Jaesik Han, Sarah Miller, and Daby Sow. Question-driven design process for explainable ai user experiences. *arXiv preprint arXiv:2104.03483*, 2021.

[47] Guangtao Zeng, Wenmian Yang, Zeqian Ju, Yue Yang, Sicheng Wang, Ruisi Zhang, Meng Zhou, Jiaqi Zeng, Xiangyu Dong, Ruoyu Zhang, et al. Meddialog: A large-scale medical dialogue dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9241–9250, 2020.

[48] Hubert Baniecki, Wojciech Kretowicz, Piotr Piatyszek, Jakub Wisniewski, and Przemyslaw Biecek. dalex: Responsible machine learning with interactive explainability and fairness in python. *Journal of Machine Learning Research*, 22(214):1–7, 2021. URL `http://jmlr.org/papers/v22/20-1473.html`.

[49] David W Scott. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015.

[50] David W Scott. On optimal and data-based histograms. *Biometrika*, 66 (3):605–610, 1979.

[51] Przemyslaw Biecek and Tomasz Burzykowski. *Explanatory Model Analysis*. Chapman and Hall/CRC, New York, 2021. ISBN 9780367135591. URL `https://pbiecek.github.io/ema/`.

[52] William S Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American statistical association*, 74(368): 829–836, 1979.

[53] Michele Cappellari, Richard M. McDermid, Katherine Alatalo, Leo Blitz, Maxime Bois, Frédéric Bournaud, M. Bureau, Alison F. Crocker, Roger L. Davies, Timothy A. Davis, P. T. de Zeeuw, Pierre-Alain

Duc, Eric Emsellem, Sadegh Khochfar, Davor Krajnović, Harald Kuntschner, Raffaella Morganti, Thorsten Naab, Tom Oosterloo, Marc Sarzi, Nicholas Scott, Paolo Serra, Anne-Marie Weijmans, and Lisa M. Young. The ATLAS^{3D} project - XX. Mass-size and mass-σ distributions of early-type galaxies: bulge fraction drives kinematics, mass-to-light ratio, molecular gas fraction and stellar initial mass function. *Monthly Notices of the Royal Astronomical Society*, 432(3):1862–1893, July 2013. doi: 10.1093/mnras/stt644.

[54] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

[55] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[56] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

[57] Thibault Sellam, Dipanjan Das, and Ankur Parikh. Bleurt: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, 2020.

[58] Miruna-Adriana Clinciu, Arash Eshghi, and Helen Hastie. A study of automatic metrics for the evaluation of natural language explanations. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2376–2387, 2021.

[59] Stefan Riezler and John T Maxwell III. On some pitfalls in automatic evaluation and significance testing for mt. In *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 57–64, 2005.

[60] Alexander Yeh. More accurate tests for the statistical significance of result differences. In *COLING 2000 Volume 2: The 18th International Conference on Computational Linguistics*, 2000.

[61] Vladimir I Levenshtein et al. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union, 1966.

Appendix A

Notes on classifiers training

A.1 Cardiovascular disease dataset

A.1.1 Random forest

For the random forest classifier we compare the 10-fold CV results on different number of estimators, they are reported in Table A.1. Considering them we keep the model with 50 estimators.

A.1.2 XGBoost

For the XGBoost classifier we do a small grid search on two of its various hyperparameters, the one which influence more the bias/variance tradeoff: the *max depth* of the tree and the *min child weight*, i.e. the minimum number of samples in a node of the tree. We consider the values $[3, 6, 9]$ for the first one (where 6 is the default) and $[1, 3, 5]$ for the second one (where 1 is the

Number of estimators	**CV Accuracy**
10	0.679
30	0.685
50	**0.688**
100	**0.688**

Table A.1: Comparison of different numbers of estimators for RF classifier on cardio dataset

MaxDepth	MinChildWeight	CV Acc
3	1	**0,735**
3	3	**0,735**
3	5	**0,735**
6	1	0,732
6	3	0,733
6	5	0,733
9	1	0,726
9	3	0,728
9	5	0,729

Table A.2: Grid-search for XGBoost on cardio dataset

default). The results with 10-fold CV are in Table A.2. We use the (3,1) version.

A.1.3 Logistic Regression

For logistic regression we compare L2 penalty, L1 penalty and no penalty. For the L1 and L2 penalty we also search the value of C in the set $[0.01, 0.1, 1, 10, 100]$. It results that the model does not show any difference with all these configurations, as shown in Table A.3. We keep the version with L2 penalty and C=1.

A.1.4 Feed Forward Neural Network (FFNN)

For the FFNN we consider a two layers network and we try different numbers of units in the layers. The network is trained with Adam optimizer, using early stopping with patience=3 to determine the number of epochs. For the learning rate we compare the results with 10^{-3} and 10^{-4}. The 10-fold CV results are reported in Table A.4. Considering them we keep the model with 10 units, trained with initial learning rate = 10^{-4}.

Penalty	**C**	**CV Acc**
L2	0,01	0,728
L2	0,1	0,728
L2	1	0,728
L2	10	0,728
L2	100	0,728
L2	1000	0,728
L1	0,01	0,727
L1	0,1	0,728
L1	1	0,728
L1	10	0,728
L1	100	0,728
L1	1000	0,728
none	n.a.	0,728

Table A.3: Grid-search for logistic regression classifier on cardio dataset

N. units	**LR**	**CV Accuracy**
10	10^{-3}	0.733
25	10^{-3}	0.734
50	10^{-3}	0.734
10	10^{-4}	**0.735**
25	10^{-4}	**0.735**
50	10^{-4}	0.731

Table A.4: Comparison of different numbers of units and lr for FFNN classifier on cardio dataset

Number of estimators	CV Accuracy
10	0.751
30	**0.766**
50	**0.766**
100	0.758

Table A.5: Comparison of different numbers of estimators for RF classifier on diabetes dataset

MaxDepth	MinChildWeight	CV Acc
3	1	**0,750**
3	3	0,729
3	5	0,734
6	1	0,747
6	3	0,729
6	5	0,736
9	1	**0.750**
9	3	**0,750**
9	5	0,728

Table A.6: Grid-search for XGBoost on diabetes dataset

A.2 Pima diabetes dataset

A.2.1 Random forest

The same comparison for the diabetes dataset is reported in Table A.5. Considering it we select the model with 30 estimators.

A.2.2 XGBoost

For XGBoost we repeat the same grid search and we report the results in Table A.6. We keep the (3,1) version.

A.2.3 Logistic Regression

We do the same grid search of the cardio dataset, with the results reported in Table A.7. We select the version without penalty.

Penalty	**C**	**CV Acc**
L2	0,01	0,758
L2	0,1	0,766
L2	1	0,767
L2	10	**0,770**
L2	100	**0,770**
L2	1000	**0,770**
L1	0,01	0,721
L1	0,1	0,768
L1	1	0,768
L1	10	0,768
L1	100	**0,770**
L1	1000	**0,770**
none	n.a.	**0,770**

Table A.7: Grid-search for logistic regression classifier on diabetes dataset

A.2.4 Feed Forward Neural Network (FFNN)

The same comparison for the diabetes dataset is reported in Table A.8. Considering it we choose the model with 10 units, trained with initial learning rate = 10^{-4}.

N. units	**LR**	**CV Accuracy**
10	10^{-3}	0.758
25	10^{-3}	0.760
50	10^{-3}	0.757
10	10^{-4}	**0.763**
25	10^{-4}	0.755
50	10^{-4}	0.759

Table A.8: Comparison of different numbers of units and learning rates for FFNN classifier on diabetes dataset

Appendix B

Grid search results for GPT-2

B.1 Explanations model

We report in Table B.1 the results of the grid search over the gpt-2 generation parameters for explanations. We report also the results for greedy and beam search:

- **Greedy**: (0.408 0.587 0.675)
- **Beam (n_beams=2)**: (0.401 0.586 0.678)
- **Beam (n_beams=5)**: (0.395 0.577 0.671)

Table B.1: Grid search for GPT-2 generation parameter

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-4	METEOR	BLEURT
1	0	10	0,9	0,7	**0,443**	**0,621**	0,680
1	0	50	0,9	0,7	0,439	0,617	0,681
1	0	100	0,9	0,7	0,440	0,618	**0,682**
1	0	1000	0,9	0,7	0,442	0,619	0,681
1	0	10	0,93	0,7	0,440	0,619	0,680
1	0	50	0,93	0,7	0,440	0,619	0,680
1	0	100	0,93	0,7	0,439	0,619	0,680
1	0	1000	0,93	0,7	0,440	0,618	0,680
1	0	10	0,96	0,7	0,437	0,616	0,679
1	0	50	0,96	0,7	0,442	0,619	0,680

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-4	METEOR	BLEURT
1	0	100	0,96	0,7	0,439	0,617	0,680
1	0	1000	0,96	0,7	0,439	0,618	0,680
1	0	10	0,99	0,7	0,437	0,614	0,676
1	0	50	0,99	0,7	0,440	0,618	0,678
1	0	100	0,99	0,7	0,440	0,616	0,677
1	0	1000	0,99	0,7	0,437	0,616	0,678
1	0	10	0,9	0,8	0,442	0,619	0,678
1	0	50	0,9	0,8	0,439	0,617	0,681
1	0	100	0,9	0,8	0,438	0,617	0,679
1	0	1000	0,9	0,8	0,441	0,618	0,680
1	0	10	0,93	0,8	0,436	0,615	0,679
1	0	50	0,93	0,8	0,436	0,614	0,678
1	0	100	0,93	0,8	0,442	0,619	0,678
1	0	1000	0,93	0,8	0,440	0,618	0,681
1	0	10	0,96	0,8	0,437	0,615	0,677
1	0	50	0,96	0,8	0,436	0,613	0,677
1	0	100	0,96	0,8	0,439	0,615	0,678
1	0	1000	0,96	0,8	0,435	0,612	0,678
1	0	10	0,99	0,8	0,435	0,613	0,675
1	0	50	0,99	0,8	0,437	0,614	0,676
1	0	100	0,99	0,8	0,434	0,611	0,676
1	0	1000	0,99	0,8	0,437	0,614	0,675
1	0	10	0,9	0,9	0,436	0,614	0,678
1	0	50	0,9	0,9	0,436	0,612	0,677
1	0	100	0,9	0,9	0,440	0,618	0,678
1	0	1000	0,9	0,9	0,435	0,612	0,677
1	0	10	0,93	0,9	0,436	0,614	0,677
1	0	50	0,93	0,9	0,436	0,613	0,676
1	0	100	0,93	0,9	0,438	0,616	0,676
1	0	1000	0,93	0,9	0,434	0,611	0,676
1	0	10	0,96	0,9	0,435	0,612	0,674

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-4	METEOR	BLEURT
1	0	50	0,96	0,9	0,432	0,611	0,674
1	0	100	0,96	0,9	0,434	0,612	0,674
1	0	1000	0,96	0,9	0,439	0,615	0,676
1	0	10	0,99	0,9	0,433	0,609	0,673
1	0	50	0,99	0,9	0,431	0,609	0,673
1	0	100	0,99	0,9	0,433	0,609	0,673
1	0	1000	0,99	0,9	0,431	0,609	0,674
1	1	10	0,9	0,7	0,438	0,618	0,680
1	1	50	0,9	0,7	0,437	0,618	**0,682**
1	1	100	0,9	0,7	0,439	0,619	**0,682**
1	1	1000	0,9	0,7	0,442	0,621	0,681
1	1	10	0,93	0,7	0,440	0,618	0,680
1	1	50	0,93	0,7	0,439	0,618	0,679
1	1	100	0,93	0,7	0,441	0,619	**0,682**
1	1	1000	0,93	0,7	**0,443**	**0,621**	0,681
1	1	10	0,96	0,7	0,439	0,617	0,679
1	1	50	0,96	0,7	0,438	0,617	0,680
1	1	100	0,96	0,7	0,442	0,619	0,680
1	1	1000	0,96	0,7	0,436	0,615	0,681
1	1	10	0,99	0,7	0,437	0,615	0,677
1	1	50	0,99	0,7	0,437	0,615	0,678
1	1	100	0,99	0,7	0,443	0,620	0,679
1	1	1000	0,99	0,7	0,439	0,616	0,679
1	1	10	0,9	0,8	**0,443**	0,620	0,679
1	1	50	0,9	0,8	0,437	0,615	0,678
1	1	100	0,9	0,8	0,438	0,616	0,679
1	1	1000	0,9	0,8	0,438	0,616	0,681
1	1	10	0,93	0,8	0,438	0,615	0,678
1	1	50	0,93	0,8	0,439	0,616	0,677
1	1	100	0,93	0,8	0,435	0,613	0,678
1	1	1000	0,93	0,8	0,438	0,615	0,679

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-4	METEOR	BLEURT
1	1	10	0,96	0,8	0,438	0,616	0,677
1	1	50	0,96	0,8	0,436	0,614	0,677
1	1	100	0,96	0,8	0,438	0,615	0,677
1	1	1000	0,96	0,8	0,439	0,617	0,679
1	1	10	0,99	0,8	0,434	0,612	0,674
1	1	50	0,99	0,8	0,435	0,612	0,675
1	1	100	0,99	0,8	0,438	0,615	0,675
1	1	1000	0,99	0,8	0,437	0,613	0,677
1	1	10	0,9	0,9	0,439	0,616	0,678
1	1	50	0,9	0,9	0,438	0,615	0,678
1	1	100	0,9	0,9	0,440	0,616	0,679
1	1	1000	0,9	0,9	0,441	0,618	0,678
1	1	10	0,93	0,9	0,436	0,613	0,675
1	1	50	0,93	0,9	0,434	0,612	0,676
1	1	100	0,93	0,9	0,441	0,617	0,677
1	1	1000	0,93	0,9	0,435	0,612	0,676
1	1	10	0,96	0,9	0,435	0,612	0,674
1	1	50	0,96	0,9	0,436	0,614	0,674
1	1	100	0,96	0,9	0,435	0,612	0,675
1	1	1000	0,96	0,9	0,435	0,612	0,676
1	1	10	0,99	0,9	0,435	0,612	0,672
1	1	50	0,99	0,9	0,434	0,612	0,672
1	1	100	0,99	0,9	0,436	0,612	0,672
1	1	1000	0,99	0,9	0,433	0,610	0,674
1	2	10	0,9	0,7	0,439	0,619	0,681
1	2	50	0,9	0,7	0,440	0,618	0,681
1	2	100	0,9	0,7	0,441	0,619	**0,682**
1	2	1000	0,9	0,7	0,439	0,618	**0,682**
1	2	10	0,93	0,7	0,441	0,620	0,680
1	2	50	0,93	0,7	0,438	0,616	**0,682**
1	2	100	0,93	0,7	0,439	0,618	0,680

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-4	METEOR	BLEURT
1	2	1000	0,93	0,7	0,438	0,617	0,679
1	2	10	0,96	0,7	0,437	0,615	0,678
1	2	50	0,96	0,7	0,440	0,618	0,679
1	2	100	0,96	0,7	0,440	0,618	0,681
1	2	1000	0,96	0,7	0,438	0,616	0,680
1	2	10	0,99	0,7	0,436	0,615	0,676
1	2	50	0,99	0,7	0,439	0,616	0,678
1	2	100	0,99	0,7	0,436	0,616	0,677
1	2	1000	0,99	0,7	0,436	0,615	0,678
1	2	10	0,9	0,8	0,441	0,619	0,679
1	2	50	0,9	0,8	0,439	0,617	0,679
1	2	100	0,9	0,8	0,439	0,618	0,680
1	2	1000	0,9	0,8	0,439	0,617	0,679
1	2	10	0,93	0,8	0,438	0,616	0,678
1	2	50	0,93	0,8	0,438	0,615	0,678
1	2	100	0,93	0,8	0,436	0,613	0,679
1	2	1000	0,93	0,8	0,437	0,615	0,679
1	2	10	0,96	0,8	0,435	0,614	0,678
1	2	50	0,96	0,8	0,438	0,616	0,678
1	2	100	0,96	0,8	0,435	0,613	0,678
1	2	1000	0,96	0,8	0,440	0,616	0,677
1	2	10	0,99	0,8	0,438	0,614	0,674
1	2	50	0,99	0,8	0,434	0,612	0,674
1	2	100	0,99	0,8	0,436	0,613	0,676
1	2	1000	0,99	0,8	0,433	0,612	0,675
1	2	10	0,9	0,9	0,436	0,613	0,677
1	2	50	0,9	0,9	0,436	0,614	0,679
1	2	100	0,9	0,9	0,438	0,615	0,678
1	2	1000	0,9	0,9	0,437	0,615	0,677
1	2	10	0,93	0,9	0,440	0,617	0,674
1	2	50	0,93	0,9	0,433	0,610	0,675

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-4	METEOR	BLEURT
1	2	100	0,93	0,9	0,439	0,615	0,676
1	2	1000	0,93	0,9	0,440	0,617	0,677
1	2	10	0,96	0,9	0,437	0,614	0,676
1	2	50	0,96	0,9	0,436	0,612	0,676
1	2	100	0,96	0,9	0,434	0,611	0,674
1	2	1000	0,96	0,9	0,435	0,611	0,674
1	2	10	0,99	0,9	0,434	0,611	0,671
1	2	50	0,99	0,9	0,433	0,611	0,672
1	2	100	0,99	0,9	0,433	0,610	0,675
1	2	1000	0,99	0,9	0,436	0,612	0,674
1,5	0	10	0,9	0,7	0,324	0,517	0,577
1,5	0	50	0,9	0,7	0,326	0,519	0,576
1,5	0	100	0,9	0,7	0,322	0,514	0,575
1,5	0	1000	0,9	0,7	0,325	0,516	0,576
1,5	0	10	0,93	0,7	0,324	0,517	0,576
1,5	0	50	0,93	0,7	0,328	0,520	0,577
1,5	0	100	0,93	0,7	0,323	0,517	0,576
1,5	0	1000	0,93	0,7	0,323	0,514	0,574
1,5	0	10	0,96	0,7	0,327	0,519	0,575
1,5	0	50	0,96	0,7	0,324	0,516	0,574
1,5	0	100	0,96	0,7	0,325	0,517	0,575
1,5	0	1000	0,96	0,7	0,322	0,515	0,574
1,5	0	10	0,99	0,7	0,323	0,514	0,573
1,5	0	50	0,99	0,7	0,322	0,515	0,573
1,5	0	100	0,99	0,7	0,320	0,514	0,573
1,5	0	1000	0,99	0,7	0,320	0,512	0,571
1,5	0	10	0,9	0,8	0,325	0,517	0,576
1,5	0	50	0,9	0,8	0,320	0,512	0,574
1,5	0	100	0,9	0,8	0,324	0,516	0,574
1,5	0	1000	0,9	0,8	0,326	0,517	0,576
1,5	0	10	0,93	0,8	0,321	0,514	0,574

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-4	METEOR	BLEURT
1,5	0	50	0,93	0,8	0,317	0,510	0,573
1,5	0	100	0,93	0,8	0,318	0,512	0,573
1,5	0	1000	0,93	0,8	0,321	0,513	0,575
1,5	0	10	0,96	0,8	0,321	0,513	0,572
1,5	0	50	0,96	0,8	0,320	0,513	0,572
1,5	0	100	0,96	0,8	0,321	0,514	0,572
1,5	0	1000	0,96	0,8	0,318	0,511	0,572
1,5	0	10	0,99	0,8	0,318	0,509	0,571
1,5	0	50	0,99	0,8	0,319	0,512	0,572
1,5	0	100	0,99	0,8	0,320	0,512	0,569
1,5	0	1000	0,99	0,8	0,319	0,512	0,570
1,5	0	10	0,9	0,9	0,320	0,513	0,573
1,5	0	50	0,9	0,9	0,317	0,510	0,574
1,5	0	100	0,9	0,9	0,319	0,513	0,573
1,5	0	1000	0,9	0,9	0,317	0,509	0,571
1,5	0	10	0,93	0,9	0,320	0,514	0,573
1,5	0	50	0,93	0,9	0,317	0,510	0,571
1,5	0	100	0,93	0,9	0,316	0,510	0,570
1,5	0	1000	0,93	0,9	0,318	0,512	0,571
1,5	0	10	0,96	0,9	0,320	0,513	0,570
1,5	0	50	0,96	0,9	0,315	0,509	0,570
1,5	0	100	0,96	0,9	0,315	0,509	0,570
1,5	0	1000	0,96	0,9	0,317	0,511	0,569
1,5	0	10	0,99	0,9	0,314	0,507	0,569
1,5	0	50	0,99	0,9	0,317	0,509	0,566
1,5	0	100	0,99	0,9	0,313	0,505	0,567
1,5	0	1000	0,99	0,9	0,313	0,505	0,566
1,5	1	10	0,9	0,7	0,324	0,515	0,575
1,5	1	50	0,9	0,7	0,323	0,515	0,574
1,5	1	100	0,9	0,7	0,322	0,517	0,574
1,5	1	1000	0,9	0,7	0,328	0,519	0,576

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-4	METEOR	BLEURT
1,5	1	10	0,93	0,7	0,323	0,515	0,575
1,5	1	50	0,93	0,7	0,324	0,516	0,575
1,5	1	100	0,93	0,7	0,323	0,514	0,576
1,5	1	1000	0,93	0,7	0,320	0,513	0,574
1,5	1	10	0,96	0,7	0,326	0,517	0,576
1,5	1	50	0,96	0,7	0,323	0,516	0,575
1,5	1	100	0,96	0,7	0,324	0,516	0,573
1,5	1	1000	0,96	0,7	0,324	0,516	0,573
1,5	1	10	0,99	0,7	0,323	0,516	0,572
1,5	1	50	0,99	0,7	0,318	0,512	0,574
1,5	1	100	0,99	0,7	0,319	0,512	0,572
1,5	1	1000	0,99	0,7	0,321	0,513	0,573
1,5	1	10	0,9	0,8	0,322	0,515	0,575
1,5	1	50	0,9	0,8	0,323	0,515	0,575
1,5	1	100	0,9	0,8	0,323	0,516	0,574
1,5	1	1000	0,9	0,8	0,323	0,517	0,575
1,5	1	10	0,93	0,8	0,323	0,515	0,574
1,5	1	50	0,93	0,8	0,321	0,515	0,574
1,5	1	100	0,93	0,8	0,322	0,515	0,575
1,5	1	1000	0,93	0,8	0,323	0,517	0,573
1,5	1	10	0,96	0,8	0,322	0,514	0,573
1,5	1	50	0,96	0,8	0,320	0,512	0,573
1,5	1	100	0,96	0,8	0,320	0,513	0,573
1,5	1	1000	0,96	0,8	0,318	0,510	0,572
1,5	1	10	0,99	0,8	0,316	0,509	0,571
1,5	1	50	0,99	0,8	0,319	0,513	0,570
1,5	1	100	0,99	0,8	0,318	0,511	0,571
1,5	1	1000	0,99	0,8	0,320	0,512	0,571
1,5	1	10	0,9	0,9	0,320	0,514	0,573
1,5	1	50	0,9	0,9	0,319	0,513	0,573
1,5	1	100	0,9	0,9	0,320	0,513	0,572

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-4	METEOR	BLEURT
1,5	1	1000	0,9	0,9	0,316	0,510	0,573
1,5	1	10	0,93	0,9	0,318	0,511	0,571
1,5	1	50	0,93	0,9	0,317	0,511	0,571
1,5	1	100	0,93	0,9	0,322	0,515	0,573
1,5	1	1000	0,93	0,9	0,320	0,513	0,572
1,5	1	10	0,96	0,9	0,318	0,511	0,570
1,5	1	50	0,96	0,9	0,316	0,510	0,571
1,5	1	100	0,96	0,9	0,319	0,514	0,572
1,5	1	1000	0,96	0,9	0,318	0,511	0,569
1,5	1	10	0,99	0,9	0,318	0,511	0,569
1,5	1	50	0,99	0,9	0,314	0,507	0,567
1,5	1	100	0,99	0,9	0,315	0,509	0,568
1,5	1	1000	0,99	0,9	0,316	0,509	0,567
1,5	2	10	0,9	0,7	0,325	0,516	0,576
1,5	2	50	0,9	0,7	0,326	0,520	0,575
1,5	2	100	0,9	0,7	0,320	0,515	0,574
1,5	2	1000	0,9	0,7	0,323	0,515	0,576
1,5	2	10	0,93	0,7	0,321	0,513	0,574
1,5	2	50	0,93	0,7	0,321	0,513	0,574
1,5	2	100	0,93	0,7	0,323	0,515	0,576
1,5	2	1000	0,93	0,7	0,323	0,514	0,575
1,5	2	10	0,96	0,7	0,323	0,515	0,575
1,5	2	50	0,96	0,7	0,324	0,517	0,574
1,5	2	100	0,96	0,7	0,321	0,514	0,575
1,5	2	1000	0,96	0,7	0,323	0,516	0,574
1,5	2	10	0,99	0,7	0,321	0,515	0,574
1,5	2	50	0,99	0,7	0,321	0,515	0,575
1,5	2	100	0,99	0,7	0,320	0,513	0,572
1,5	2	1000	0,99	0,7	0,322	0,514	0,573
1,5	2	10	0,9	0,8	0,322	0,516	0,575
1,5	2	50	0,9	0,8	0,324	0,518	0,575

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-4	METEOR	BLEURT
1,5	2	100	0,9	0,8	0,321	0,514	0,574
1,5	2	1000	0,9	0,8	0,323	0,516	0,575
1,5	2	10	0,93	0,8	0,322	0,515	0,575
1,5	2	50	0,93	0,8	0,322	0,515	0,571
1,5	2	100	0,93	0,8	0,320	0,512	0,574
1,5	2	1000	0,93	0,8	0,324	0,515	0,575
1,5	2	10	0,96	0,8	0,320	0,514	0,574
1,5	2	50	0,96	0,8	0,322	0,514	0,571
1,5	2	100	0,96	0,8	0,319	0,513	0,573
1,5	2	1000	0,96	0,8	0,318	0,512	0,573
1,5	2	10	0,99	0,8	0,323	0,516	0,570
1,5	2	50	0,99	0,8	0,317	0,510	0,571
1,5	2	100	0,99	0,8	0,318	0,511	0,570
1,5	2	1000	0,99	0,8	0,319	0,512	0,571
1,5	2	10	0,9	0,9	0,322	0,514	0,571
1,5	2	50	0,9	0,9	0,320	0,513	0,574
1,5	2	100	0,9	0,9	0,322	0,515	0,573
1,5	2	1000	0,9	0,9	0,316	0,511	0,573
1,5	2	10	0,93	0,9	0,321	0,515	0,574
1,5	2	50	0,93	0,9	0,318	0,511	0,572
1,5	2	100	0,93	0,9	0,320	0,514	0,572
1,5	2	1000	0,93	0,9	0,318	0,513	0,571
1,5	2	10	0,96	0,9	0,315	0,509	0,570
1,5	2	50	0,96	0,9	0,317	0,508	0,570
1,5	2	100	0,96	0,9	0,317	0,510	0,568
1,5	2	1000	0,96	0,9	0,316	0,509	0,570
1,5	2	10	0,99	0,9	0,318	0,511	0,568
1,5	2	50	0,99	0,9	0,319	0,512	0,569
1,5	2	100	0,99	0,9	0,317	0,507	0,566
1,5	2	1000	0,99	0,9	0,315	0,507	0,566

B.2 Q&A model

We report in Table B.2 the results of the grid search over the gpt-2 generation parameters for the Q&A. None of these parameters combinations for sampling is better than the **greedy search**, which results in (**0.190**, **0.453**, **0.574**)

- **Beam (n_beams=2)**: (0.189, 0.447, 0.570)
- **Beam (n_beams=5)**: (0.169, 0.417, 0.572)

Table B.2: Grid search for GPT-2 generation parameters for Q&A

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-1	METEOR	BLEURT
1	0	10	0,9	0,7	0,165	0,409	0,520
1	0	50	0,9	0,7	0,163	0,412	0,520
1	0	100	0,9	0,7	0,164	0,412	0,521
1	0	1000	0,9	0,7	0,170	0,415	0,523
1	0	10	0,93	0,7	0,169	0,415	0,521
1	0	50	0,93	0,7	0,167	0,415	0,521
1	0	100	0,93	0,7	0,165	0,411	0,518
1	0	1000	0,93	0,7	0,165	0,411	0,519
1	0	10	0,96	0,7	0,160	0,405	0,515
1	0	50	0,96	0,7	0,170	0,417	0,521
1	0	100	0,96	0,7	0,156	0,404	0,516
1	0	1000	0,96	0,7	0,167	0,410	0,521
1	0	10	0,99	0,7	0,159	0,407	0,516
1	0	50	0,99	0,7	0,155	0,404	0,515
1	0	100	0,99	0,7	0,163	0,409	0,518
1	0	1000	0,99	0,7	0,162	0,407	0,516
1	0	10	0,9	0,8	0,157	0,404	0,514
1	0	50	0,9	0,8	0,162	0,409	0,516
1	0	100	0,9	0,8	0,170	0,412	0,521
1	0	1000	0,9	0,8	0,161	0,404	0,515
1	0	10	0,93	0,8	0,158	0,402	0,514
1	0	50	0,93	0,8	0,164	0,410	0,518

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-1	METEOR	BLEURT
1	0	100	0,93	0,8	0,159	0,403	0,515
1	0	1000	0,93	0,8	0,163	0,409	0,518
1	0	10	0,96	0,8	0,156	0,402	0,511
1	0	50	0,96	0,8	0,156	0,400	0,511
1	0	100	0,96	0,8	0,159	0,404	0,517
1	0	1000	0,96	0,8	0,161	0,406	0,515
1	0	10	0,99	0,8	0,162	0,406	0,516
1	0	50	0,99	0,8	0,156	0,400	0,514
1	0	100	0,99	0,8	0,156	0,400	0,511
1	0	1000	0,99	0,8	0,154	0,399	0,512
1	0	10	0,9	0,9	0,165	0,407	0,515
1	0	50	0,9	0,9	0,162	0,406	0,515
1	0	100	0,9	0,9	0,157	0,402	0,510
1	0	1000	0,9	0,9	0,162	0,402	0,512
1	0	10	0,93	0,9	0,150	0,395	0,508
1	0	50	0,93	0,9	0,168	0,412	0,516
1	0	100	0,93	0,9	0,152	0,397	0,510
1	0	1000	0,93	0,9	0,158	0,399	0,512
1	0	10	0,96	0,9	0,149	0,399	0,511
1	0	50	0,96	0,9	0,150	0,395	0,506
1	0	100	0,96	0,9	0,152	0,394	0,508
1	0	1000	0,96	0,9	0,150	0,397	0,509
1	0	10	0,99	0,9	0,144	0,390	0,507
1	0	50	0,99	0,9	0,149	0,396	0,511
1	0	100	0,99	0,9	0,153	0,399	0,511
1	0	1000	0,99	0,9	0,144	0,391	0,508
1	1	10	0,9	0,7	0,195	0,417	0,546
1	1	50	0,9	0,7	0,189	0,415	0,545
1	1	100	0,9	0,7	0,195	0,417	0,547
1	1	1000	0,9	0,7	0,190	0,422	0,548
1	1	10	0,93	0,7	0,185	0,410	0,543

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-1	METEOR	BLEURT
1	1	50	0,93	0,7	0,101	0,419	0,548
1	1	100	0,93	0,7	0,195	0,415	0,546
1	1	1000	0,93	0,7	0,191	0,414	0,545
1	1	10	0,96	0,7	**0,199**	0,420	0,547
1	1	50	0,96	0,7	0,190	**0,424**	0,547
1	1	100	0,96	0,7	0,188	0,415	0,543
1	1	1000	0,96	0,7	0,197	0,416	0,546
1	1	10	0,99	0,7	0,191	0,416	0,544
1	1	50	0,99	0,7	0,188	0,414	0,544
1	1	100	0,99	0,7	0,196	0,419	0,545
1	1	1000	0,99	0,7	0,188	0,412	0,543
1	1	10	0,9	0,8	0,188	0,412	0,543
1	1	50	0,9	0,8	0,198	0,417	0,546
1	1	100	0,9	0,8	0,191	0,413	0,545
1	1	1000	0,9	0,8	0,188	0,409	0,543
1	1	10	0,93	0,8	0,193	0,415	0,545
1	1	50	0,93	0,8	0,195	0,416	0,545
1	1	100	0,93	0,8	0,184	0,419	0,547
1	1	1000	0,93	0,8	0,194	0,412	0,545
1	1	10	0,96	0,8	0,188	0,411	0,544
1	1	50	0,96	0,8	0,194	0,416	0,545
1	1	100	0,96	0,8	0,187	0,410	0,543
1	1	1000	0,96	0,8	0,190	0,411	0,544
1	1	10	0,99	0,8	0,190	0,412	0,544
1	1	50	0,99	0,8	0,188	0,412	0,541
1	1	100	0,99	0,8	0,192	0,415	0,543
1	1	1000	0,99	0,8	0,194	0,416	0,544
1	1	10	0,9	0,9	0,197	0,414	0,545
1	1	50	0,9	0,9	0,194	0,415	0,546
1	1	100	0,9	0,9	0,190	0,418	0,547
1	1	1000	0,9	0,9	0,191	0,415	0,546

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-1	METEOR	BLEURT
1	1	10	0,93	0,9	0,187	0,410	0,543
1	1	50	0,93	0,9	0,190	0,420	0,546
1	1	100	0,93	0,9	0,193	0,414	0,545
1	1	1000	0,93	0,9	0,190	0,408	0,544
1	1	10	0,96	0,9	0,187	0,408	0,542
1	1	50	0,96	0,9	0,185	0,407	0,541
1	1	100	0,96	0,9	0,191	0,409	0,543
1	1	1000	0,96	0,9	0,185	0,408	0,542
1	1	10	0,99	0,9	0,193	0,410	0,544
1	1	50	0,99	0,9	0,196	0,416	0,544
1	1	100	0,99	0,9	0,196	0,416	0,547
1	1	1000	0,99	0,9	0,186	0,411	0,543
1	2	10	0,9	0,7	0,167	0,413	0,521
1	2	50	0,9	0,7	0,164	0,410	0,522
1	2	100	0,9	0,7	0,167	0,410	0,521
1	2	1000	0,9	0,7	0,163	0,409	0,520
1	2	10	0,93	0,7	0,163	0,413	0,519
1	2	50	0,93	0,7	0,166	0,412	0,521
1	2	100	0,93	0,7	0,162	0,408	0,517
1	2	1000	0,93	0,7	0,162	0,410	0,520
1	2	10	0,96	0,7	0,165	0,411	0,517
1	2	50	0,96	0,7	0,169	0,412	0,521
1	2	100	0,96	0,7	0,156	0,403	0,519
1	2	1000	0,96	0,7	0,156	0,407	0,515
1	2	10	0,99	0,7	0,161	0,406	0,517
1	2	50	0,99	0,7	0,162	0,409	0,518
1	2	100	0,99	0,7	0,160	0,408	0,516
1	2	1000	0,99	0,7	0,163	0,410	0,518
1	2	10	0,9	0,8	0,168	0,411	0,519
1	2	50	0,9	0,8	0,159	0,405	0,516
1	2	100	0,9	0,8	0,172	0,415	0,519

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-1	METEOR	BLEURT
1	2	1000	0,9	0,8	0,160	0,408	0,517
1	2	10	0,93	0,8	0,159	0,404	0,515
1	2	50	0,93	0,8	0,167	0,412	0,518
1	2	100	0,93	0,8	0,160	0,403	0,514
1	2	1000	0,93	0,8	0,159	0,401	0,514
1	2	10	0,96	0,8	0,158	0,404	0,514
1	2	50	0,96	0,8	0,153	0,402	0,513
1	2	100	0,96	0,8	0,155	0,405	0,514
1	2	1000	0,96	0,8	0,164	0,403	0,514
1	2	10	0,99	0,8	0,154	0,399	0,511
1	2	50	0,99	0,8	0,155	0,405	0,514
1	2	100	0,99	0,8	0,161	0,404	0,514
1	2	1000	0,99	0,8	0,159	0,404	0,514
1	2	10	0,9	0,9	0,155	0,400	0,510
1	2	50	0,9	0,9	0,159	0,401	0,513
1	2	100	0,9	0,9	0,159	0,401	0,513
1	2	1000	0,9	0,9	0,167	0,410	0,517
1	2	10	0,93	0,9	0,150	0,398	0,511
1	2	50	0,93	0,9	0,151	0,400	0,510
1	2	100	0,93	0,9	0,163	0,404	0,514
1	2	1000	0,93	0,9	0,158	0,401	0,512
1	2	10	0,96	0,9	0,146	0,392	0,508
1	2	50	0,96	0,9	0,148	0,397	0,508
1	2	100	0,96	0,9	0,155	0,398	0,511
1	2	1000	0,96	0,9	0,147	0,395	0,509
1	2	10	0,99	0,9	0,160	0,404	0,512
1	2	50	0,99	0,9	0,148	0,396	0,507
1	2	100	0,99	0,9	0,151	0,395	0,510
1	2	1000	0,99	0,9	0,152	0,395	0,510
1,5	0	10	0,9	0,7	0,141	0,408	0,561
1,5	0	50	0,9	0,7	0,139	0,404	0,560

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-1	METEOR	BLEURT
1,5	0	100	0,9	0,7	0,143	0,407	0,559
1,5	0	1000	0,9	0,7	0,144	0,409	0,559
1,5	0	10	0,93	0,7	0,138	0,401	0,560
1,5	0	50	0,93	0,7	0,140	0,404	0,559
1,5	0	100	0,93	0,7	0,139	0,407	0,561
1,5	0	1000	0,93	0,7	0,140	0,405	0,557
1,5	0	10	0,96	0,7	0,138	0,404	0,560
1,5	0	50	0,96	0,7	0,137	0,404	0,559
1,5	0	100	0,96	0,7	0,140	0,406	0,559
1,5	0	1000	0,96	0,7	0,137	0,405	0,559
1,5	0	10	0,99	0,7	0,138	0,407	0,557
1,5	0	50	0,99	0,7	0,140	0,409	0,555
1,5	0	100	0,99	0,7	0,137	0,405	0,557
1,5	0	1000	0,99	0,7	0,136	0,404	0,557
1,5	0	10	0,9	0,8	0,136	0,403	0,558
1,5	0	50	0,9	0,8	0,136	0,403	0,558
1,5	0	100	0,9	0,8	0,137	0,404	0,556
1,5	0	1000	0,9	0,8	0,137	0,405	0,558
1,5	0	10	0,93	0,8	0,133	0,402	0,555
1,5	0	50	0,93	0,8	0,137	0,405	0,556
1,5	0	100	0,93	0,8	0,133	0,401	0,555
1,5	0	1000	0,93	0,8	0,137	0,405	0,555
1,5	0	10	0,96	0,8	0,133	0,402	0,555
1,5	0	50	0,96	0,8	0,135	0,404	0,555
1,5	0	100	0,96	0,8	0,137	0,405	0,556
1,5	0	1000	0,96	0,8	0,137	0,405	0,555
1,5	0	10	0,99	0,8	0,135	0,406	0,554
1,5	0	50	0,99	0,8	0,133	0,403	0,554
1,5	0	100	0,99	0,8	0,137	0,405	0,555
1,5	0	1000	0,99	0,8	0,134	0,403	0,554
1,5	0	10	0,9	0,9	0,135	0,403	0,555

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-1	METEOR	BLEURT
1,5	0	50	0,9	0,9	0,130	0,401	0,553
1,5	0	100	0,9	0,9	0,134	0,403	0,554
1,5	0	1000	0,9	0,9	0,134	0,403	0,554
1,5	0	10	0,93	0,9	0,132	0,401	0,553
1,5	0	50	0,93	0,9	0,129	0,400	0,553
1,5	0	100	0,93	0,9	0,133	0,403	0,553
1,5	0	1000	0,93	0,9	0,133	0,404	0,554
1,5	0	10	0,96	0,9	0,130	0,402	0,553
1,5	0	50	0,96	0,9	0,131	0,403	0,552
1,5	0	100	0,96	0,9	0,132	0,402	0,552
1,5	0	1000	0,96	0,9	0,132	0,405	0,554
1,5	0	10	0,99	0,9	0,133	0,402	0,551
1,5	0	50	0,99	0,9	0,131	0,402	0,552
1,5	0	100	0,99	0,9	0,132	0,403	0,552
1,5	0	1000	0,99	0,9	0,129	0,401	0,551
1,5	1	10	0,9	0,7	0,124	0,418	0,566
1,5	1	50	0,9	0,7	0,124	0,416	0,565
1,5	1	100	0,9	0,7	0,120	0,412	0,566
1,5	1	1000	0,9	0,7	0,127	0,418	0,566
1,5	1	10	0,93	0,7	0,125	0,416	0,566
1,5	1	50	0,93	0,7	0,128	0,421	0,565
1,5	1	100	0,93	0,7	0,123	0,415	0,566
1,5	1	1000	0,93	0,7	0,121	0,414	0,564
1,5	1	10	0,96	0,7	0,126	0,418	0,565
1,5	1	50	0,96	0,7	0,124	0,416	0,564
1,5	1	100	0,96	0,7	0,119	0,412	0,563
1,5	1	1000	0,96	0,7	0,125	0,416	0,565
1,5	1	10	0,99	0,7	0,122	0,416	0,563
1,5	1	50	0,99	0,7	0,118	0,412	0,563
1,5	1	100	0,99	0,7	0,124	0,416	0,563
1,5	1	1000	0,99	0,7	0,121	0,414	0,563

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-1	METEOR	BLEURT
1,5	1	10	0,9	0,8	0,119	0,412	0,564
1,5	1	50	0,9	0,8	0,125	0,416	0,566
1,5	1	100	0,9	0,8	0,120	0,415	0,565
1,5	1	1000	0,9	0,8	0,125	0,418	0,565
1,5	1	10	0,93	0,8	0,120	0,414	0,565
1,5	1	50	0,93	0,8	0,126	0,418	0,565
1,5	1	100	0,93	0,8	0,122	0,414	0,563
1,5	1	1000	0,93	0,8	0,122	0,414	0,565
1,5	1	10	0,96	0,8	0,121	0,415	0,562
1,5	1	50	0,96	0,8	0,123	0,416	0,563
1,5	1	100	0,96	0,8	0,122	0,413	0,563
1,5	1	1000	0,96	0,8	0,118	0,411	0,562
1,5	1	10	0,99	0,8	0,121	0,414	0,562
1,5	1	50	0,99	0,8	0,120	0,413	0,561
1,5	1	100	0,99	0,8	0,119	0,411	0,561
1,5	1	1000	0,99	0,8	0,118	0,411	0,560
1,5	1	10	0,9	0,9	0,119	0,413	0,562
1,5	1	50	0,9	0,9	0,121	0,415	0,563
1,5	1	100	0,9	0,9	0,121	0,414	0,564
1,5	1	1000	0,9	0,9	0,121	0,413	0,561
1,5	1	10	0,93	0,9	0,120	0,413	0,561
1,5	1	50	0,93	0,9	0,118	0,411	0,560
1,5	1	100	0,93	0,9	0,119	0,412	0,561
1,5	1	1000	0,93	0,9	0,120	0,412	0,561
1,5	1	10	0,96	0,9	0,117	0,409	0,561
1,5	1	50	0,96	0,9	0,119	0,413	0,560
1,5	1	100	0,96	0,9	0,116	0,408	0,560
1,5	1	1000	0,96	0,9	0,117	0,409	0,561
1,5	1	10	0,99	0,9	0,117	0,408	0,560
1,5	1	50	0,99	0,9	0,115	0,409	**0,569**
1,5	1	100	0,99	0,9	0,116	0,409	0,568
1,5	1	1000	0,99	0,9	0,118	0,410	0,566

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-1	METEOR	BLEURT
1,5	2	10	0,9	0,7	0,138	0,411	0,505
1,5	2	50	0,9	0,7	0,142	0,412	0,507
1,5	2	100	0,9	0,7	0,143	0,416	0,504
1,5	2	1000	0,9	0,7	0,144	0,412	0,505
1,5	2	10	0,93	0,7	0,138	0,405	0,504
1,5	2	50	0,93	0,7	0,136	0,405	0,501
1,5	2	100	0,93	0,7	0,140	0,411	0,503
1,5	2	1000	0,93	0,7	0,138	0,409	0,503
1,5	2	10	0,96	0,7	0,138	0,408	0,502
1,5	2	50	0,96	0,7	0,139	0,409	0,502
1,5	2	100	0,96	0,7	0,134	0,403	0,502
1,5	2	1000	0,96	0,7	0,142	0,412	0,503
1,5	2	10	0,99	0,7	0,135	0,408	0,501
1,5	2	50	0,99	0,7	0,135	0,408	0,501
1,5	2	100	0,99	0,7	0,140	0,415	0,503
1,5	2	1000	0,99	0,7	0,136	0,408	0,502
1,5	2	10	0,9	0,8	0,134	0,405	0,501
1,5	2	50	0,9	0,8	0,137	0,409	0,501
1,5	2	100	0,9	0,8	0,137	0,411	0,502
1,5	2	1000	0,9	0,8	0,135	0,408	0,499
1,5	2	10	0,93	0,8	0,136	0,410	0,501
1,5	2	50	0,93	0,8	0,131	0,406	0,498
1,5	2	100	0,93	0,8	0,137	0,409	0,500
1,5	2	1000	0,93	0,8	0,136	0,411	0,498
1,5	2	10	0,96	0,8	0,132	0,407	0,499
1,5	2	50	0,96	0,8	0,134	0,409	0,499
1,5	2	100	0,96	0,8	0,129	0,402	0,495
1,5	2	1000	0,96	0,8	0,133	0,404	0,500
1,5	2	10	0,99	0,8	0,135	0,412	0,499
1,5	2	50	0,99	0,8	0,132	0,407	0,497
1,5	2	100	0,99	0,8	0,127	0,402	0,496

(continue)

Rep Pen	NRNGS	Topk	TopP	Temp	BLEU-1	METEOR	BLEURT
1,5	2	1000	0,99	0,8	0,132	0,407	0,497
1,5	2	10	0,9	0,9	0,132	0,404	0,496
1,5	2	50	0,9	0,9	0,139	0,411	0,499
1,5	2	100	0,9	0,9	0,127	0,403	0,497
1,5	2	1000	0,9	0,9	0,133	0,405	0,497
1,5	2	10	0,93	0,9	0,131	0,402	0,496
1,5	2	50	0,93	0,9	0,136	0,409	0,498
1,5	2	100	0,93	0,9	0,134	0,409	0,497
1,5	2	1000	0,93	0,9	0,131	0,407	0,497
1,5	2	10	0,96	0,9	0,128	0,403	0,495
1,5	2	50	0,96	0,9	0,127	0,405	0,496
1,5	2	100	0,96	0,9	0,125	0,402	0,493
1,5	2	1000	0,96	0,9	0,127	0,404	0,493
1,5	2	10	0,99	0,9	0,127	0,403	0,495
1,5	2	50	0,99	0,9	0,128	0,406	0,492
1,5	2	100	0,99	0,9	0,130	0,409	0,493
1,5	2	1000	0,99	0,9	0,129	0,400	0,495