Master in Artificial Intelligence 2022-23

Explainable and Trustworthy AI

I1+I2+I3 Answer Report

Pedro Guijas Bravo

February 14, 2023

Contents

1 Introduction

This report will answer the questions posed in Notebooks 1, 2 and 3. Therefore, it should be noted that only elements such as code or graphics that are not in the Jupyters Notebooks will be included.

Additionally, the lack of clarity and modularity of the code, as well as the use of libraries with a very poor documentation quality (SimpleNLG) should be highlighted. This, as well as multiple errors encountered have hindered the development of the practice (see an example below).

Listing 1: error example

```
iris_value_inf = irisClass_ind.getValue()

# but using iris_inf_value (instead of iris_value_inf) to print class name
print(" (OUTPUT): " + str(irisClass_ind.getName()) + "=" + str(iris_value_inf) + "
    ("+iris_class_names[round(iris_inf_value)-1]+")")
```

2 Notebook 1

2.1 Exercise 1

- **Which is the inferred output class by each fuzzy system?**
  - **Mamdani FRBS**:
    - Sample 7 (iris-setosa): Class=1.0 (Setosa)
    - Sample 85 (iris-versicolor): Class=2.0 (Versicolor)
    - Sample 135 (iris-virginica): Class=3.0 (Virginica)
  - **TSK FRBS**:
    - Sample 7 (iris-setosa): Class=1.0 (Setosa)
    - Sample 85 (iris-versicolor): Class=2.0 (Versicolor)
    - Sample 135 (iris-virginica): Class=3.0 (Virginica)
- **Which are the fired rules?** For this and future questions, it is important to note that the number immediately after the name of the rule will be the degree of activation of the rule.
  - **Mamdani FRBS**
    - Sample 7 (iris-setosa):
      - **rule1 -** (0.9166666) IF Petal Width IS low THEN Class IS 1.0
      - **rule2 -** (0.08333334) IF Petal Length IS more or less (average) AND Petal Width IS average THEN Class IS 2.0
    - 85 (iris-versicolor):
      - **rule2 -** (0.74999994) IF Petal Length IS more or less (average) AND Petal Width IS average THEN Class IS 2.0
      - **rule3 -** (0.18644066) IF Petal Length IS high AND Petal Width IS average THEN Class IS 3.0
      - **rule4 -** (0.25000006) IF Petal Width IS high THEN Class IS 3.0
    - 135 (iris-virginica):
      - **rule2 -** (0.1666667) IF Petal Length IS more or less (average) AND Petal Width IS average THEN Class IS 2.0
      - **rule3 -** (0.1666667) IF Petal Length IS high AND Petal Width IS average THEN Class IS 3.0

      - **rule4** - (0.8333333) IF Petal Width IS high THEN Class IS 3.0

  - **TSK FRBS**
    - Sample 7 (iris-setosa):
      - **rule1** - (0.9166666) IF Petal Width IS low THEN Class IS Setosa
      - **rule2** - (0.08333334) IF Petal Length IS low AND Petal Width IS average THEN Class IS Versicolor
    - 85 (iris-versicolor):
      - **rule2** - (0.6101695) IF Petal Length IS low AND Petal Width IS average THEN Class IS Versicolor [weight=1.0]
      - **rule3** - (0.13983048) IF Petal Length IS high AND Petal Width IS average THEN Class IS Virginica [weight=1.0]
      - **rule4** - (0.25000006) IF Petal Width IS high THEN Class IS Virginica [weight=1.0]
    - 135 (iris-virginica):
      - **rule2** - (0.04519776) IF Petal Length IS low AND Petal Width IS average THEN Class IS Versicolor
      - **rule3** - (0.12146894) IF Petal Length IS high AND Petal Width IS average THEN Class IS Virginica
      - **rule4** - (0.8333333) IF Petal Width IS high THEN Class IS Virginica

- **Pay attention to the firing degrees: Are they the same in Mamdani and TSK? (if not, explain why)**

  They are not the same. This is due to how the activation degrees are computed. In Mamdani, for given terms joined with an *and* operator, the activation of the rule will be computed by obtaining the minimum degree of membership, for the *or* operation it will be by computing the maximum. Note that the output will be defuzzyfied by computing the centroid. However, in this case, only the rule with the highest degree of activation will be identified.

  For Sugeno, again the activation of a rule depends on the *and* and *or* terms are computed using t-Norm, i.e. computing products. Focusing on the *and* operators, as can be intuited, this operation will be more restrictive, since when multiplying elements less than 1 (degrees of membership) the result will be less than or equal to the minimum (Mamdani). Therefore, in Sugeno it is smoothed more and it must be emphasized that the output is not computed with the centroid, instead it is computed with a function.

2.2 Exercise 2

- **Which is the inferred output class by each fuzzy system?**
  - **Real class**: Class=2.0 (Versicolor)
  - **Fuzzy System with Regular Partitions**: Class=2.0 (Versicolor)
  - **Fuzzy System with Induced Partitions**: Class=2.0 (Versicolor)

- **Which are the fired rules?**
  - **Fuzzy System with Regular Partitions**:
    - **rule3** - (0.6440678) IF Petal Length IS average AND Petal Width IS average THEN Class IS 2.0
    - **rule4** - (0.35593218) IF Petal Length IS high AND Petal Width IS average THEN Class IS 3.0
    - **rule5** - (0.3333334) IF Petal Width IS high THEN Class IS 3.0
  - **Fuzzy System with Induced Partitions**:
    - **rule3** - (0.50746256) IF Petal Length IS average AND Petal Width IS average THEN Class IS 2.0

    * **rule4** - (0.49253744) IF Petal Length IS average AND Petal Width IS high THEN Class IS 3.0
    * **rule5** - (0.40746042) IF Petal Length IS high THEN Class IS 3.0

- **Are they the same with regular or induced partitions? (if not, explain why)**

  Obviously they are not the same, since at the time of fuzzyfication, the terms of the variables are defined on other intervals, causing that the membership functions and the values they return are not the same given input variables. Therefore, since the activation of the rule depends on the degrees of membership, different degrees of activation are obtained for the same rule in each fuzzy system.

- **Can you write an explanation of the output given by each system (in terms of the fired rules)?**

  First, we identify the rules with the highest degree of activation for each system:

  - **Fuzzy System with Regular Partitions**: **rule3** - (0.6440678) IF Petal Length IS average AND Petal Width IS average THEN Class IS 2.0
  - **Fuzzy System with Induced Partitions**: **rule3** - (0.50746256) IF Petal Length IS average AND Petal Width IS average THEN Class IS 2.0

  According to these rules, the most likely and feasible explanation will be the same for both systems, with the difference that the degree of confidence of the system with induced partitions will be clearly lower. Therefore the class is probably 2 (Versicolor), being the decision based on the fact that the Petal Length is in the average, as well as the Petal Width.

  However, rules 4 and 5 indicate that it may also be possible (although less likely) that the sample belongs to class 3 (Vigilica). For the system with induced partitions it is clearly more likely (activations of 0.49 and 0.40 versus 0.35 and 0.33 for both systems respectively).

  Rule 4 indicates that it could happen that with a normal Petal Length and high Petal Width the sample could be of class 3 (Vigilica). And similarly, rule 5 indicates that with Petal Length high the sample could be of the same class.

  Finally, the values of the attributes are shown so that the reader can judge for himself the membership values and degrees of activation of the systems.

  - **Sepal Length**: 6.7
  - **Sepal Width**: 3.0
  - **Petal Length**: 5.0
  - **Petal Width**: 1.7

2.3 Exercise 3

- **Which is the most accurate system for each dataset?**

  - **Wine**: **Random Forest**, with an average classification ratio of 0.98 and a standard deviation of 0.01. It is the most accurate and stable model.
  - **Beer**: **Random Forest**, with a mean classification ratio of 0.97 and a standard deviation of 0.02. Again the most stable and accurate model.

- **Which is the most interpretable system for each dataset?**

  The simpler it is to understand the inner workings of a system, the more interpretable it is. Therefore, simpler systems will be easier to understand. The simplicity of the model will be defined by the number of rules for fuzzy systems and the number of leaves for decision trees.

  - **Wine**: The simplest model will be the **Decision Tree**, with 9 leaves and a standard deviation of 2.65. However, the fuzzy system **SP-WM-S**, with a mean of 9.4 rules and a standard deviation of 2.51, should be mentioned.

  - **Beer**: **RP-FDTP-S** is the simplest model, with a mean of 8.4 active rules and a standard deviation of 0.52

- **Which is the system with the best interpretability-accuracy trade-off for each dataset?**

  Using the Pareto front we will try to identify the model closest to the optimal point for each dataset. Note that the optimal point will be the lowest number of rules and the perfect accuracy.

  Note that interpretability has not been calculated for the Random Forest and, consequently, will not be treated. It is assumed that this is because it is very complicated to understand the decision motivation of the model.

  - **Wine**: Multiple models have a very similar distance, these models are **Decision Tree**, **RP-FDTP-S** and **SP-WM-S**.
  - **Beer**: Clearly, the most balanced model will be **RP-FDTP-S**.

3 Notebook 2

3.1 Exercise 1

- **Which output class is inferred by the fuzzy system?**
  - **Real class**: Class=1.0 (Blanche)
  - **RP_FDTP_S**: Class=1.0 (Blanche)
  - **FURIA with linguistic approximation a priori**: Class=1.0 (Blanche)
  - **FURIA with post-hoc linguistic approximation**: Class=1.0 (Blanche)
- **Why? (Which are the fired rules? Which is the related factual explanation?)**
  - **RP_FDTP_S**:
    * **rule1** - (0.6666667) IF Color IS Pale AND Bitterness IS Low THEN Class IS Blanche
    * **rule2** - (0.10330578) IF Color IS Pale AND Bitterness IS High THEN Class IS Pilsner
    * **rule4** - (0.31578958) IF Color IS Straw AND Bitterness IS Low AND Strength IS Standard THEN Class IS Lager

    Factual explanation: The test instance is of class blanche because color is pale and bitterness is low.
  - **FURIA with linguistic approximation a priori**:
    * **rule1** - (1.0) IF Bitterness IS Low AND Color IS Pale or Straw THEN Class IS Blanche
    * **rule2** - (1.0) IF Bitterness IS Low or Low-medium AND Color IS Pale or Straw THEN Class IS Blanche

    Factual explanation: The test instance is of class blanche because bitterness is low and color is pale or straw.
  - **FURIA with post-hoc linguistic approximation**:
    * **rule1** - (1.0) IF Bitterness IS MF0 AND Color IS MF0 THEN Class IS Blanche
    * **rule2** - (1.0) IF Bitterness IS MF1 AND Color IS MF1 THEN Class IS Blanche

    Factual explanation: The test instance is of class blanche because bitterness is low and color is pale.
- **Why the selected data instance is not classified as Lager?**

- **RP_FDTP_S**: The test instance would be of class lager if color were straw and strength were standard.

  That is, for the activation of the rule that determines the class as Larger to be selected, the values of color and strength should change, specifically they should increase. Thus, by increasing the values of these variables, the membership function of the Staw and Standar terms would increase, increasing the activation of the rule and overcoming the rest.

- **FURIA with linguistic approximation a priori**: The test instance would be of class lager if bitterness were low or low-medium or medium-high and color were amber.

  The reasoning is the same as for the fuzzy system **RP_FDTP_S**, but in this case, only the color variable should increase to increase the amber membership function. A more detailed explanation will be given in the next question.

- **FURIA with post-hoc linguistic approximation**: The test instance would be of class lager if color were amber.

  In this case, not having the appropriate linguistic labels in the fuzzy system, it is impossible to perform the reasoning. Since we have a counterfactual explanation using one terminology and a factual explanation and fuzzy system using another (and in the code, the rule base is not shown, only Blanche rules). If it would at least indicate the amber membership function, some reasoning might be possible.

• **What is the minimal change in features (Color, Bitterness, Strength) that is required to flip the classification into Lager?**

- **RP_FDTP_S**: As mentioned, color must increment to a value greater than 4.5 (2.5 increment), where the term of the varaible color with the highest membership value is Straw.

  As for the Strength variable, following the same reasoning, it will have to increase its value sufficiently so that Rule 1 is not the one with the highest degree of activation. That is, since the only change so far has been to decrease the Pale term in favor of Straw in the color variable, the Standard term in the Strength variable must slightly exceed 0.5 in its membership value so as not to penalize this increase in the execution of the rule. Remember that the activation of the rule with ands for Mamdani depends on the minimum membership value. Therefore, the Stength variable should reach a value of 0.0485.

  Since the explanation can be confusing, the need for the increment in the Strength variable is attached visually in Figure 1.

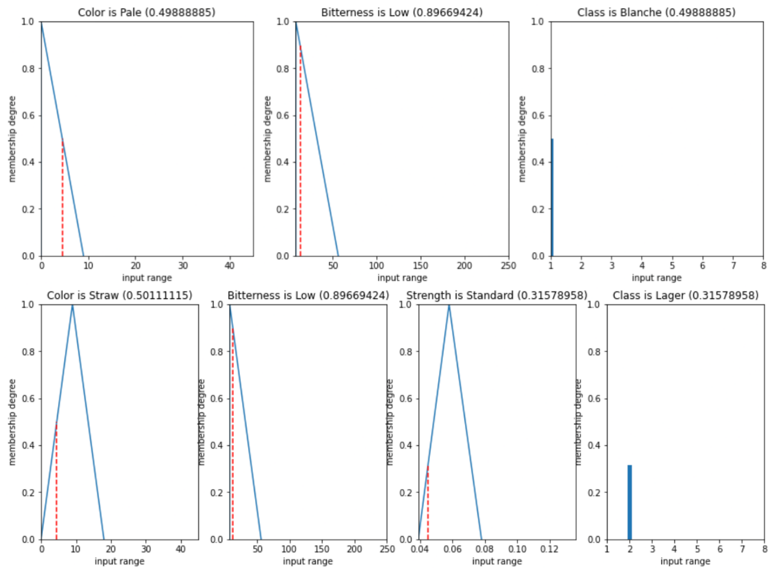


Figure 1: Penalization of the Standard term in the Strength variable for the rule that determines the output as Large.

  - **FURIA with linguistic approximation a priori**: Following the previously mentioned contractual explanation, the values of the variables bitterness and color should be modified. For the color to be majorly amber, it should be increased to a value of 8 (where it will already exceed straw), an increase of 5. Additionally, bitterness should be low, low-medium or medium-high, with its current value of 13 being adequate.
  - **FURIA with post-hoc linguistic approximation**: As mentioned in the previous question, there is no way to obtain the correlation between amber and the mfX terms. Additionally, the global set of rules for this system is not printed, being impossible to manually relate the terms. Thus, as the question only indicates to execute (not to code), it has not been possible to obtain an answer.

3.2 Exercise 2

- **How many numerical and categorical features are in the dataset?** The dataset consists of 14 features, one of which is the value to be predicted (like) and is categorical. As for the rest, some variables whose domain is integers can be interpreted as categorical (as indicated by their semantics in the dataset). Therefore, these variables will be: key, like and mode. The result is 10 numerical features, 3 categorical features and 1 categorical output.
- **Are classes well balanced?** Yes, you have approximately the same number of instances for each class.
- **Do you identify missing values or outliers?** The distributions of values show no anomalies. The 0's in the dataset are valid values, not missing data. Also, no outliers have been found either.

3.3 Exercise 3

- **Which is the most accurate classifier? Random Forest** is the model with the best accuracy, with a 0.73 mean and only a 0.03 standard deviation.
- **Which is the most interpretable classifier? FURIA_W** is the most interpretable model because it has the lowest average number of active rules, with an average number of 3.54. That is, it is the simplest model and consequently the easiest to interpret.
- **Which is the classifier with the best interpretability-accuracy trade-off?** Again, as in the previous Notebook the interpretability of the Random Forest will not be computed. Thus, the model with the best interpretability-accuracy trade-off will be **FURIA_W** (the explanation is analogous to exercise 2.3).

3.4 Exercise 4

- **Do you appreciate any difference between the generated histograms and the information previously discovered with Google Facets?** No, only the bin size is larger in Google Facets.
- **Do you think linguistic variables are properly defined with meaningful linguistic terms?** Of course, the use of terms such as very low, low... are very familiar and highly used in natural language, which makes them easy to understand.
- **How many rules are in each rule base?**
  - **RP_FDTP_S**: 80
  - **FURIA**: 12
- **How many rules are fired for the two data instances that are taken as inputs?**

  - **RP_FDTP_S**:
    * **Instance 642**: 5
    * **Instance 1920**: 10
  - **FURIA**:
    * **Instance 642**: 2
    * **Instance 1920**: 1

- **What about the generated factual and counterfactual explanations? (Are they consistent? What are the minimal changes needed to flip the output?)**

  First of all, the instances were well classified, being the factual explanations the following:

  - **RP_FDTP_S**:
    * **Instance 642**: The test instance is of class 2.0 because energy is (average) or (average high) or (high) or (very high), instrumentalness is very low and loudness is average high.
    * **Instance 1920**: The test instance is of class 1.0 because danceability is high, loudness is very high and tempo is (low) or (average low).
  - **FURIA**:
    * **Instance 642**: The test instance is of class 2.0 because duration is medium, instrumentalness is low and valence is medium.
    * **Instance 1920**: The test instance is of class 1.0 because loudness is high.

  The explanations are much more concise for the **FURIA** model, treating the explanations of the **RP_FDTP_S** model as more variables and values.

  Additionally, these explanations are not very consistent regarding the use of features. However, they can be extracted by comparing the importance of common variables such as instrumentalness for sample 642, where both agree that it should be low. Similarly, for sample 1920 they agree that loudness should generally be high.

  As for minimal changes, to change the output first we will focus on the counterfactual explanations:

  - **RP_FDTP_S**:
    * **Instance 642**: The test instance would be of class 1.0 if danceability were high, duration were low, loudness were high and speechiness were very low.
    * **Instance 1920**: The test instance would be of class 2.0 if danceability were average high or high or very high, instrumentalness were very low, loudness were high and speechiness were low or average low or average.
  - **FURIA**:
    * **Instance 642**: The test instance would be of class 1.0 if speechiness were low, instrumentalness were *(nothing, it is inferred by looking at the rules that should display MF1)* and valence were high.

      For this explanation, it should be noted that the code does not return an attribute value.
    * **Instance 1920**: The test instance would be of class 2.0 if duration were medium, instrumentalness were low and valence were medium.

  As for the minimum changes, they are as follows:

  - **RP_FDTP_S**: The method to calculate them will be the same as the one used in a previous similar exercise: taking as reference the counterfactual explanations and looking at the distribution of the terms. The minimum point for each term will be calculated.

    In this case, the way to calculate it will be to calculate the midpoint of a segment (degree of membership of 0.5) since only triangular and trapezoidal transfer functions are involved.

In this distribution of terms specifically the intersections with other terms cut right at 0.5, having the certainty that other terms for that variable will be strictly smaller and guaranteeing the minimum upper degrees of membership for the corresponding rule to be executed.

Therefore, the minimal changes will be as follows:

- **Instance 642**:
  - **danceability (0.84)**: It is high with sufficient confidence, it is not modified.
  - **duration (632693)**: Decrement to approximately 263188.255 to be low with a confidence greater than 0.5.
  - **loudness (-9.003)**: Up to -8.5045 to have confidence in high term greater than 0.5.
  - **speechiness (0.0391)**: Its current value is adequate given that it is already a very low with a confidence of 0.878.
- **Instance 1920**:
  - **danceability (0.821)**: It is in correct values (average high, high or very high).
  - **instrumentalness (1.4e-05)**: It is in correct values (very low).
  - **loudness (-9.003)**: Should increase slightly to -8.5045 (high)
  - **speechiness (0.0391)**: Should be increased to 0.089 (low, medium low or medium).

– **FURIA**: Using the same reasoning as in exercise 1 of this same notebook, it is really complicated to calculate the minimum changes. First of all, the linguistic labels do not help to know the semantics of the term (being again different from those of the counterfactual explanation). And secondly, although it is not possible to relate a term of the counterfactual explanation to a term of a variable, they have an extremely inhomogeneous distribution. See the example of the instrumental variable, where the different terms have the same membership functions and their distribution complicates the explainability of the model:

- MF0 - trapezoid [a: 0.0, b: 0.0, c: 0.0, d: 0.0]
- MF1 - trapezoid [a: 0.0, b: 0.0, c: 0.0, d: 0.0]
- MF2 - trapezoid [a: 0.0, b: 0.0, c: 0.98, d: 0.98]
- MF3 - trapezoid [a: 0.011, b: 0.011, c: 0.98, d: 0.98]
- MF4 - trapezoid [a: 0.0, b: 0.0, c: 0.98, d: 0.98]
- MF5 - trapezoid [a: 0.0, b: 0.0, c: 0.98, d: 0.98]

3.5 Exercise 5

- **Compare the given classifications and related explanations for the two data instances under consideration in the SONGS dataset. Higlight advantages and disadvantages (but also identify similarity and difference) of each method. In case of detecting inconsistencies, discuss why they turn up.**

  – **RP_FDTP_S vs dtc5**: The methods to be compared are RP_FDTP_S, a fuzzy rule-based system, and dtc5, a decision tree which limits its maximum depth to 5.

  Focusing on the given classifications, both methods have similar values. However, the decision tree is clearly superior with a precision of 0.699 and higher stability (low standard deviation: 0.007). On the other hand, the rule-based system has a precision of 0.66 and an associated standard deviation of 0.13.

  In terms of interpretability, the rule-based system has 59 activated rules on average, while the decision tree has only 28 leaves. This means that the tree is a simpler model and consequently easier to interpret. Therefore, bearing in mind the accuracy and interpretability of the models, the decision tree with depth limit is considered clearly superior.

  As for the operation of both methods, they have a common part, since it is the existence of rules. In the case of decision trees, these are formed taking into account the routes to reach each leaf.

However, the main difference of these methods is the activation of these rules. The fuzzy system employs a discretization stage in terms for a given variable, having membership values for each one, this being an intrinsic characteristic of fuzzy logic. Thus, the rule that determines the output will be the one with the highest degree of activation (dependent on the degrees of membership of the variables). In decision trees the process is simpler, in each node a question about an attribute will be asked, so that the concatenation of questions will guide the process to an output.

Finally, it should be noted that for both instances (642 and 1920) the fuzzy model manages to correctly classify both. On the other hand, the model with a priori the best performance fails in the 642 intrance, classifying it as dislike. This failure is due to the depth limitation, since the unconstrained decision tree is able to classify it well. Although the generalization capacity increases with the depth constraint since it avoids overfitting, the leaves will have a clearly higher entropy than the model without pruning, resulting in the misclassification of some very specific cases. Note that this is visible to the naked eye in the plotting of such trees.

– **InterpretML vs SHAP**: We will proceed to compare InterpretML, specifically the ExplainableBoostingClassifier model versus SHAP, a method for explaining the results of any automatic learning model based on game theory. In this way, the advantage and disadvantage of each method can already be intuited a priori: the specification of the first one and the generalization of the second one.

  Comparing the global importance of the terms, both models coincide, highlighting the great importance of the instrumentalness attribute, followed by loudness. However, in the prediction of instances they differ. For instance 642 the attribute that most conditions the result for InterpretML is danceability and valence. On the contrary for SHAP with XGB it is duration. For the second sample something similar happens, the most determining attribute is intrumentalness and loudness for the first model and only loudness for the second one.

- **Do you agree with the given classifications and explanations (keeping in mind that the two songs under consideration are given below)?**

  – **642: Get Lucky - Daft Punk Remix, Daft Punk(like → 2): 0.0267,0.821,632693, 0.793,1.37e-05,6,0.0751,-9.003,0,0.0391,116.009,4.0,0.91,2**

  – **1920: Remember the Time, Michael Jackson (dislike → 1): 0.153,0.831,239227 ,0.921,0.00213,5,0.305,-2.383,0,0.0581,108.002,4.0,0.808,1**

  First of all, the label given by the dataset for the first sample does not seem to me to be entirely appropriate, being the feeling generated more of indifference than dislike.

  Regarding the first sample, I completely agree with InterpretML's explanation that the main reason for liking is the danceability and valence or loudness of the song. But regarding the second model, even if I have the previous characteristics in mind I completely differ with that one of the major ranking motives is the length of the song.

  Regarding the second sample, the first model justifies the output mainly with the instrumentalness characteristic. That is, the song is less vocal, accompanying a base that although I don't dislike it, I don't love it either. Finally, the second model bases its response essentially on the loudness attribute, which causes my disagreement, because similarly to the duration, it does not seem to me that this decision can be justified solely on the basis of this value.

4 Notebook 3

4.1 Exercise 1

- **For the data instances given below with indexes [3, 78, 135] in the wine dataset, check which is the output predicted by the two models previously generated (clf3 and clf5). In addition, use the SimpleNLG library which was introduced in session**

I2 to automatically generate a sentence explaining each classification output. With that aim, for the sake of modularity, code a Python function that automatically generates such an explanation.

Code another function taken as input the whole tree, and the given data input. Then, you have to traverse and parse the whole tree in order to identify the fired rule, and later you can call to the function previously implemented for getting the linguistic explanation.

Below, the first code segment shows the function given a rule and an example shows an explanation using SimpleNLG. In addition, code execution with an example is included. Two implementation details should be noted:

– The use of dictionaries to identify variables. This decision has been motivated by the flexibility and ease they offer, as opposed to the operation with indexes.

– Although the input sample is available and the difference could be calculated to explain in natural language whether these are (very) large or (very) small (for example: hue (0.5) is **much** lower than 0.94), it has been decided not to use these adjectives. This is because the differences are relative to the domain of the variables. For a probability bounded between 0 and 1 a difference of 0.5 will be large, whereas for a probability domain bounded between 0 and 100 it will be minuscule. Given the need to include information on the domain, as well as the fact that it is not explicitly requested, it has been preferred not to run the risk of including modifying adjectives that may be incorrect.

The result of the execution of this first code segment is as follows: *The Class is 2 because proline (750) is lower than 755.00, od280/od315_of_diluted_wines (2) is lower than 2.11, hue (0.5) is lower than 0.94 and flavanoids (1.25) is lower than 1.58.*

Listing 2: Python example

```python
# -------------- Function definition --------------
def explanationGenerator(fired_rule, data_input):
    # Split Rule
    then_split = fired_rule.split("THEN class:")
    output_class = then_split[1]
    # Split conditions
    conditions = [term.split() for term in then_split[0].split("&&")]
    # Defensive Programming
    assert len(then_split)==2
    for cond in conditions:
      assert len(cond)==3

    # CLASS
    class_phrase = nlgFactory.createClause()
    subj1 = nlgFactory.createNounPhrase("the", "class")
    class_phrase.setSubject(subj1)
    verb1= nlgFactory.createVerbPhrase("be")
    class_phrase.setVerb(verb1)
    class_phrase.setFeature(Feature.TENSE, Tense.PRESENT)
    obj1 = nlgFactory.createNounPhrase(output_class) # class
    class_phrase.setObject(obj1)

    # Build sentence
    sentence = nlgFactory.createCoordinatedPhrase()
    sentence.addCoordinate(class_phrase)

    first = True

    # Features
    for (x,s,y) in conditions:
        # phrase elements
        obj1 = nlgFactory.createNounPhrase(x + f" ({data_input[x]})")
        adj = nlgFactory.createAdjectivePhrase("high" if s == ">" else "low")
        adj.setFeature(Feature.IS_COMPARATIVE, True)
        than = nlgFactory.createPrepositionPhrase("than")
        adj.addComplement(than)
        obj2 = nlgFactory.createNounPhrase(y)
        than.addComplement(obj2)

        # sentence
        sent = nlgFactory.createClause()
        sent.setSubject(obj1)
        sent.setVerb("be")
        sent.setObject(adj)

        # Add explanation clause
        if first:
          sent.setFeature(Feature.COMPLEMENTISER, "because")
          class_phrase.addComplement(sent)
          first=False # only 1 because
        else:
          sent.addCoordinate(sent)

      return realiser.realiseSentence(sentence)

# -------------- Execution --------------
fired_rule = "proline <= 755.00 && od280/od315_of_diluted_wines <= 2.11 && hue
    <= 0.94 && flavanoids <= 1.58 THEN class: 2"
data_input = dict(zip(["proline","od280/od315_of_diluted_wines", "hue",
    "flavanoids"],[750, 2, 0.5, 1.25, 2]))
print(explanationGenerator(fired_rule, data_input))
```

Finally, the next code fragment shows the function that generates the decision tree paths in text format and uses the previous function to generate the explanation. The code that executes this function on the 2 models and on the 3 samples is also included. The use of feature and class names should be emphasized to make the mapping more understandable. The results will be as follows:

- **Model clf3**:
  * **Sample 3**: The Class is class_0 because proline (1480.0) is bigger than 755.0, flavanoids (3.49) is bigger than 2.165000081062317 and magnesium (113.0) is lower than 135.5.
  * **Sample 78**: The Class is class_1 because proline (750.0) is lower than 755.0, od280/od315_of_diluted_wines (2.31) is bigger than 2.1149998903274536 and flavanoids (1.85) is bigger than 0.7950000166893005.
  * **Sample 135**: The Class is class_2 because proline (695.0) is lower than 755.0, od280/od315_of_diluted_wines (1.58) is lower than 2.1149998903274536 and hue (0.73) is lower than 0.9350000023841858.
- **Model clf5**:
  * **Sample 3**: The Class is class_0 because proline (1480.0) is bigger than 755.0, flavanoids (3.49) is bigger than 2.165000081062317 and magnesium (113.0) is lower than 135.5.
  * **Sample 78**: The Class is class_1 because proline (750.0) is lower than 755.0, od280/od315_of_diluted_wines (2.31) is bigger than 2.1149998903274536, flavanoids (1.85) is bigger than 0.7950000166893005 and alcohol (12.33) is lower than 13.174999713897705.
  * **Sample 135**: The Class is class_2 because proline (695.0) is lower than 755.0, od280/od315_of_diluted_wines (1.58) is lower than 2.1149998903274536, hue (0.73) is lower than 0.9350000023841858 and flavanoids (0.66) is lower than 1.5800000429153442.

Listing 3: Python example

```python
# -------------- Function definition --------------
def get_path(dt, samples, class_names, feature_names):
  feature = dt.tree_.feature
  threshold = dt.tree_.threshold
  leaf_id = dt.apply(samples)
  dt_output = dt.predict(samples)
  node_indicator = dt.decision_path(samples)
  output = []

  for sample_id in range(len(samples)):
    node_ids = node_indicator.indices[node_indicator.indptr[sample_id] :
        node_indicator.indptr[sample_id + 1]]

    # Compute path
    str_path = ""
    for node_id in node_ids:

        # next node if leaf
        if leaf_id[sample_id] == node_id:
            continue

        # add && only if not first
        if str_path!="":
            str_path += " && "

        # check if value of the split feature for sample 0 is below threshold
        if samples[sample_id][feature[node_id]] <= threshold[node_id]:
            threshold_sign = "<="
        else:
            threshold_sign = ">"

        str_path += f"{feature_names[feature[node_id]]} {threshold_sign}
            {threshold[node_id]}"

    # Add class
    str_path += f" THEN class: {class_names[dt_output[sample_id]]}"
    # Compute and append the NL explanation
    sample_dict = dict(zip(wine.feature_names,samples[sample_id]))
    output.append(explanationGenerator(str_path, sample_dict))

  return output

# -------------- Execution --------------
samples = [X[id] for id in wine_ind]

for i,dt in zip(["clf3","clf5"],[clf3,clf5]):
  print(f"\nModel {i}:")
  for s in get_path(dt, samples, wine.target_names, wine.feature_names):
    print(f"\t {s}")
```

4.2 Exercise 2

- **Design a questionnaire with Microsoft Forms in order to evaluate the goodness of the generated textual explanations. Make public the link to the questionnaire and share it with all your classmates in the Teams channel. The form must include:**
  1. **Welcome Page**
  2. **Introduction to the survey and instructions**

3. **Select arbitrarily one of the 3 previous data instances, present the data values and the associated classification + textual explanation, followed by a Likert question, asking respondents to rate the goodness of the given explanation from 1 "very bad" to 5 "very good".**
4. **Farewell Page**

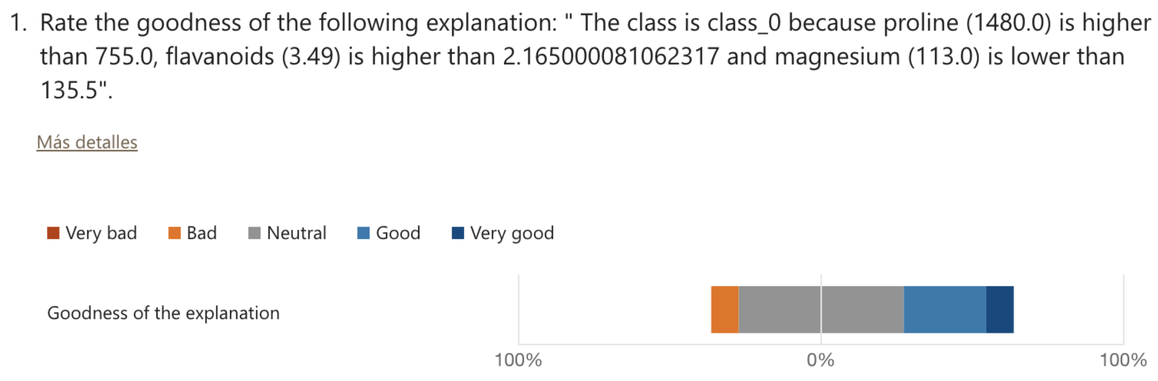
- **Your deliverable must include a copy of the form and the related analysis of responses.**

  To access the required form, please refer to the following link: Microsoft Office Forms link.

  The form has been distributed among groups of students and more, but not many responses have been collected (only 11). This means that the brief summary obtained may not be fully representative.

  The summary of the responses is shown in Figure 2. In general, the responses agree that it is neither an exaggeratedly bad nor an exaggeratedly good explanation, with the centroid of the votes being the neutral opinion.

  In general, we agree with the results obtained, being the reason for choosing the output class justified, but nevertheless the values of the characteristics do not have any justification. It is considered necessary more detail on these cut-off values, having some justification related to the minimization of entropy or the learning process, which justifies the values. Therefore, it is considered that the explanation is normal/good, being far from the best possible.



1. Rate the goodness of the following explanation: " The class is class_0 because proline (1480.0) is higher than 755.0, flavanoids (3.49) is higher than 2.165000081062317 and magnesium (113.0) is lower than 135.5".

Figure 2: Caption