International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Agile-based Requirements Engineering for Machine Learning: A Case Study on Personalized Nutrition

Carlos Cunha*[1], Rafael Oliveira[2], Rui Duarte[3]

Submitted: 18/09/2023 **Revised**: 19/11/2023 **Accepted**: 29/11/2023

Abstract: Requirements engineering is crucial in developing machine learning systems, as it establishes the foundation for successful project execution. Nevertheless, incorporating requirements engineering approaches from traditional software engineering into machine learning projects presents new challenges. These challenges arise from replacing the software logic derived from static software specifications with dynamic software logic derived from data. This paper presents a case study exploring an agile requirement engineering approach popular in traditional software projects to specify requirements in machine learning software. These requirements allow reasoning about the correctness of software and design tests for validation. The absence of software specification in machine learning software is offset by employing data quality metrics, which are assessed using cutting-edge methods for model interpretability. A case study on personalized nutrition and physical activity demonstrated the adequacy of user stories and acceptance criteria format, popular in agile projects, for specifying requirements in the machine learning domain.

Keywords: *requirements engineering, machine learning, deep learning, explainability, agile, user stories, acceptance criteria*

1. Introduction

The requirements engineering (RE) field presents distinct challenges regarding machine learning (ML) systems, which are not encountered in traditional information system development. Non-functional requirements elicitation and quality assurance of machine learning models and applications are a few examples.

Software engineers should understand ML performance measures to specify reasonable functional requirements, know quality requirements such as explainability, and integrate ML specifics in the RE process [1]. ML should also prioritize its validation process, particularly concerning RE, rather than solely focusing on verification, which examines whether the model has learned the correct specifications. Validation may determine whether the acquired behavior of an ML-based system is erroneous, even though the learning algorithm is implemented accurately.

User stories and acceptance criteria are popular software engineering artifacts for requirements specification [2]. Acceptance criteria for ML models play a crucial role in evaluating their performance and determining their suitability for deployment. These criteria are benchmarks against which the model predictions or classifications are measured, providing a quantitative or qualitative assessment of effectiveness. Additionally, they can specify data requirements to validate the quality of datasets used to define the system behavior.

Current research on ML systems must integrate with existing RE methodologies and tools [3]. One reason is the impact of ML uncertainty on requirements engineering methods. Previous solutions address ML uncertainty using goal-oriented requirements analysis (GORE) [4] or represent requirements as ML metamodels for describing the environment, system state and data, the ML behaviors, and the learning behavior [5]. There is a research gap covering the specification of ML requirements using conventional agile artifacts that stakeholders can interpret and validate. Data requirements are the core of requirements specification since they compromised the system correctness but only became available in production. Hence, datasets are system inputs that require validation because they change over time and compromise the applicational logic.

We address requirements specification using agile artifacts, as such user stories and acceptance criteria to answer the following research questions:

- RQ1. How to specify data and model performance requirements for machine learning systems using agile specification artifacts?
- RQ2. Which data quality metrics can constrain the acceptance of external data for model training?

This work presents two main contributions:

- A method that adapts well-known RE techniques from

[1] *Polytechnic Institute of Viseu, Portugal*
ORCID ID : 0000-0002-2754-5401
[2] *Polytechnic Institute of Viseu, Portugal*
[3] *Polytechnic Institute of Viseu, Portugal*
ORCID ID : 0000-0002-6819-0985
** Corresponding Author Email: cacunha@estgv.ipv.pt*

traditional software to suit machine learning projects, offering a significant scientific contribution and a practical solution for software engineering practitioners.

- The introduction of metrics for requirements specification observable through state-of-the-art model interpretability methods.

Requirements specification is analyzed under a personalized nutrition and exercise control case study. By examining specific scenarios, we can uncover the underlying factors that drive the specification of machine learning system requirements using the artifacts known in software engineering and model interpretability methods explored in state-of-the-art research.

The rest of this paper is structured to provide a clear and organized presentation of the research findings and analysis. Section II presents the state of the art on requirements engineering for machine learning. Section III defines the problem and presents the case study analyzed in this paper. Section IV revises the models and data requirements to be addressed by ML systems. Section V presents the ML requirements for the case study adopted for the analysis. Section VI discusses the results. Section VII presents the conclusions and future work.

2. Related Work

The related work on requirements engineering of ML systems covers the mapping between traditional requirements engineering approaches and the ML system requirements, validation of ML requirements, and testing approaches.

Vogelsang et al. [1] presented the perspectives of data scientists on the practices of requirements engineering in machine learning systems. They pinpointed the understanding of ML performance measures to state good functional requirements, integration of ML specifics in the RE process, and quality requirements (e.g., explainability) as the new responsibilities of requirement engineers.

Pei et al. [6] presented an overview and reflection on the collaboration among the different roles in requirements engineering for machine learning applications. Requirements engineering was proposed as a stakeholders' collaboration activity to identify data description, performance metrics, data quality, and candidate solutions.

Habibullah et al. [7] evaluated the importance of non-functional requirements (NFRs) for ML systems as perceived among practitioners from both industry and academia. Some NFRs, such as retrainability, were considered new requirements for ML systems. Some challenges arising from retrainability were the specification of when to retrain, how to retrain, and which data to use.

Nalchigar et al. [8] extended the conceptual modeling framework presented in their previous work [9] with user-story templates to elicitate elements in the business view. User stories, however, did not specify acceptance criteria to validate models and data quality.

Traditional requirements and quality attributes become less applicable when dealing with data-driven systems. Similarly, conventional requirements for fault detection techniques, like inspections, may be less effective in identifying issues in such systems. Challa et al. [10] proposed manipulating data quality characteristics by requirements engineers to achieve specific data characteristics. They validated the concept by evaluating metamorphic stationarity to preprocess the time series data to become stationary and thus make decrease RNN prediction errors.

Kuwajima et al. [11] addressed the expected performance level and evaluation criteria of automated driving systems through scenario-based verification for safety-critical machine learning systems. The authors identified gaps and areas that need further development by comparing the existing conventional system quality models, such as SQuaRE [12], with the demands of machine learning models. Machine learning models possess distinct characteristics, such as reliance on large-scale data, adaptability, and interpretability challenges.

Previous work on RE pinpoints the specification of data properties in ML systems as a fundamental activity since software behavior relies on models trained using external data. The system behavior is unknown when the requirements are specified and may change every time the model is trained or retrained. This paper uses a personalized nutrition and physical exercise case study to evaluate the suitability of user stories and acceptance criteria to describe ML requirements. The recent advancements in model interpretability methods have allowed for the specification of model properties using software engineering artifacts that are understandable to all product stakeholders.

3. Problem Definition

Requirements engineering plays a critical role in successfully developing and deploying ML software. However, several pitfalls can arise during the requirements engineering process tailored to ML projects. Awareness of these pitfalls is crucial to mitigate risks and ensure the alignment of ML systems with stakeholders' needs.

3.1. Requirements Engineering Pitfalls

Some of the common pitfalls encountered in requirements engineering for machine learning include:

- Insufficient Domain Understanding: ML systems operate within specific domains, and requirements gathering can be challenging without a deep

understanding of the domain. Inadequate domain knowledge can misinterpret stakeholders' needs, resulting in inaccurate or irrelevant requirements.

- Ambiguity in Requirement Specification: Ambiguous requirements can cause confusion and miscommunication among stakeholders. Vague or imprecise descriptions of ML model behaviors, performance expectations, or desired outputs may lead to misunderstandings during system development.
- Uncertainty and Volatility of Data: ML systems heavily rely on data, and the availability, quality, and volatility of data can pose significant challenges. Incomplete or inconsistent data can result in inaccurate requirements or bias in ML models. Changing data distributions over time may require continuous adaptation of requirements.
- Lack of Adequate Training Data: ML models require a representative and diverse dataset for training. Inadequate or biased training data can lead to suboptimal model performance or unintended behavior. Insufficient consideration of data requirements during the requirements engineering phase can hinder development.
- Poor Requirements Traceability and Management: Traceability between requirements and subsequent design decisions, implementation, and testing is essential for managing the complexity of ML projects. Insufficient traceability can lead to difficulties in maintaining and updating ML systems.

These pitfalls demand appropriate requirements specification and management.

3.2. User Stories and Acceptance Criteria

Requirements engineering on machine learning projects can benefit from adopting user stories and acceptance criteria. These two artifacts are popular in agile software development methodologies (e.g., SCRUM [2]). User stories describe high-level requirements from the business viewpoint in the form:

As [a user persona], I want [to perform this action] so that [I can accomplish this goal]

Further, the acceptance criteria describe the validation criteria of the assigned user story, which include the data properties being validated. Combining user stories with acceptance criteria provides a concise and user-centric approach to capturing requirements and ensuring that the delivered software meets stakeholders' expectations. Here is how user stories and acceptance criteria are helpful for requirements engineering:

- User-Centric Perspective: User stories focus on the end users' or stakeholders' needs and goals. They provide a narrative description of a specific user's interaction with the software system.
- Concise and Incremental Requirements: User stories are typically short and focused, capturing one specific feature or functionality at a time. They promote an incremental and iterative development approach, allowing the team to deliver value in smaller increments.
- Testable and Measurable Acceptance Criteria: Acceptance criteria define a user story's desired outcomes or behaviors. They provide a concrete and measurable way to determine when a user story is complete and meets the stakeholders' expectations. Acceptance criteria ensure that requirements are specific, unambiguous, and testable, enabling the development team to verify that the software satisfies the specified criteria.
- Improved Traceability and Requirement Management: User stories and acceptance criteria clearly link the stated requirements and the corresponding functionality. This traceability helps manage and track requirements progress throughout the development lifecycle.

Notwithstanding the specification artifacts are similar in traditional and machine projects, the software properties specified are different. Software behavior in ML projects is obtained from data, which makes requirements engineering dependent on data requirements.

3.3. Acceptance Criteria for ML

Unlike traditional functions manually designed and implemented based on domain knowledge, data-driven models can be seen as dynamic functions that learn their behavior from data examples. They capture complex patterns and make predictions or generate outputs that might be challenging to achieve with explicit functions alone.

ML software behavior can change because of concept drift changes or the availability of new data that can improve model performance. Thus, the acceptance criteria shall be validated continuously in parallel with system execution at production time.

Fig. **1** presents a traditional software development process and a machine learning software development process. In the former, the acceptance criteria specified for user stories are validated before software deployment in production. In the latter, the acceptance criteria require continuous validation after the software deployment in production since:

- data that defines the software behavior is used to train and retrain the models during the production phase;
- software behavior may become inappropriate after some time due to concept drift issues.

As model behavior depends on training external data aligned with dynamic concepts, the acceptance criteria should validate data properties instead of model behavior. We can enhance their reliability, trustworthiness, and safety by checking data properties to validate machine learning models. This approach contributes to developing more robust and accountable machine learning systems, ensuring they meet the desired properties and perform as expected in various contexts. This paper presents a case study for adopting user stories and acceptance criteria in a personalized nutrition project.

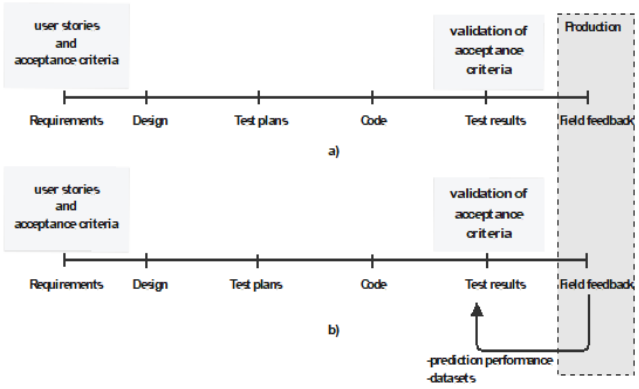

Fig. 1. Differences between a) traditional software development processes and b) machine learning software development processes

3.4. Case Study

We validate the executability of agile requirements engineering artifacts for requirements engineering on a personalized nutrition and physical activity project. Personalized nutrition is a trend that depends on sensors to collect automatically (1) input data, such as calories burned during exercise provided by smartwatches; and (2) data used by the feedback loop, such as weight, Body Mass Index (BMI), and lean mass. These data are often complemented with data introduced manually, such as the user nutrition goals and the food intake types and quantities. An intelligent control loop provided by an ML model will continuously help the user attain their goals by assisting them in choosing adequate food types and amounts.

The architecture presented in Fig. 2 describes the flow between elements. The food intake and exercise metrics represent the inputs of the system that are used to predict the BMI of a person using an ML model.

The case study involves a regression setting where the BMI of a person is predicted based on their food intake and exercise habits. This case study does not aim to present a comprehensive solution for the personalized nutrition and physical activity problem but to use it to study the application of agile requirements engineering artifacts to machine learning projects.

4. Requirements Specification in ML Systems

The performance of a machine-learning solution depends on (1) data quality metrics during the model testing stage; and (2) model prediction accuracy during the production stage.

4.1. Data Quality Metrics

Data quality is critical to the success of the model. Three essential aspects of data quality are *accuracy*, *completeness*, and *appropriateness* [13]. Fig. 3 presents the relationship between each aspect and the methods used for their evaluation.

Accuracy refers to the correctness of the data (i.e., the data is free from errors or inconsistencies). If the data is inaccurate, the model will make incorrect predictions or decisions, which can lead to negative consequences. In our infrastructure, data accuracy is limited to the accuracy of sensors.

Completeness refers to the extent to which all the data required to make accurate predictions is present. The model may not provide accurate predictions if some relevant data is missing. Missing data can also introduce bias in the model if the missing data is not random. In our work, data completeness requires validation since it is a quality attribute that depends on the availability and selection of data. Specification and testing of data completeness rely on the data distribution of features. For example, assuming that exercise intensity is a reliable predictor of BMI, it becomes essential for the samples in the training dataset to possess equivalent values to those utilized as inputs during the production stage.

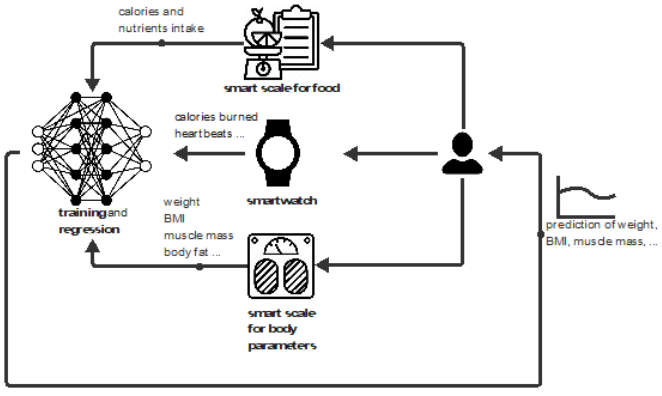

Fig. 2. System architecture for personalized nutrition

Appropriateness refers to whether the data is suitable for the intended use. In other words, the data should be relevant to the problem. The model may make accurate predictions or decisions if the data is appropriate. That means the dataset should have variables and values impacting the prediction

output. Model performance using the test dataset is a good indicator of appropriateness. At production time, the model prediction performance indicates its suitability.

4.2. Prediction Performance Metrics

Considering the regression setting of the case study, we decided on Root Mean Square Error (RMSE) to measure model performance. It is a commonly used metric in statistics and machine learning to evaluate the accuracy of a prediction model or estimator. RMSE measures the average magnitude of the differences between predicted and actual values. Equation 1 presents the formula of RMSE.

$$E = \sqrt{\frac{1}{N_v}\sum_{p=1}^{N_v}\left[\mathbf{t}_p - \mathbf{y}_p\right]^2} \quad (1)$$

In the formula, N_v represents the total number of observations, t_p is the actual observed value of the target variable for the i-th data point, and y_p is the predicted value for the i-th data point.

5. Requirements Specification Implementation

This section presents the functional and non-functional requirements for the case study addressed in this paper. The main interactions with the application respect reading data from the scales for body parameters and food, and from the smartwatch concerning heartbeats and calories burned. However, only body parameters read by the scale can provide control data to evaluate the prediction performance.

We decided on Long Short-term Memory (LSTM) networks [14] to create models since the prediction of BMI for a day also depends on the BMI observed for the previous days. We chose a publicly available dataset [15].

5.1. Functional Requirements

Listing 1 describes the user story and acceptance criteria for reading new BMI from a smart scale. It evaluates prediction performance and handles poor performance scenarios that lead to model retraining. The evaluation thresholds are specified according to specific acceptable limits for errors. The model training activity is executed when the prediction error exceeds the threshold recurrently.

Evaluation of the model performance determines whether a new model should be trained to replace the current one. Accepting a newly trained model depends on their performance evaluated with the test dataset exceeding a specific acceptability threshold and compliance with data quality requirements. The rest of this section describes the specification of data quality requirements for the case study.

5.2. Data Accuracy

The accuracy of smartwatches and smart scales depend mostly on the specification provided by the manufacturer. It may also be evaluated in the production infrastructure when a ground truth for evaluation is available. The lack of ground truth for comparison in most scenarios makes experimental evaluation error-prone, so we exclude it from the requirements specification.

5.3. Data Appropriateness

Data is appropriate if they can provide feature values to create models adjusted to the modeled phenomena. Accordingly, validation of appropriateness recurs to performance metrics for model evaluation or finer granularity evaluation at the feature level.

Listing 2 presents the user story and acceptance criteria for validating data appropriateness by specifying the threshold of acceptability for the RMSE when running the test dataset.

5.4. Data Completeness

Data requirements involve understanding the distribution of data for relevant features. Features with strong discriminating power should be provided in the training dataset with values that cover the feature space of production data homogeneously to avoid large uncovered feature space gaps.

Feature-level evaluation is performed by explainable models that expose the relationship between feature values and the model output. Traditional feature selection methods provide a list of relevant features, but their contribution to the output (i.e., their weights on the decision) demands a different class of methods. SHAP [16], LIME, and ELI5 are methods designed to explain how the input data features impact the output. We decided on the SHAP for a preliminary analysis for requirements specification. SHAP is adequate for regression problems, is well documented, and has been proven effective in several prediction scenarios [17][18][19].

Fig. 3 presents the variables identified by SHAP to justify the BMI value obtained by regression. The impact of each variable on the output is measured by the SHAP value, whose magnitude represents their impact on the output. Conversely, some features impact the output inversely to what is expected. As an example, the feature *TrackerDistance_t-2* contributes positively to BMI. Even for an expert, it is not straightforward to understand how the walking distance done three days before the prediction time

```
Feature: Reading new BMI from a smart scale
User Story: As a user, I want to read my BMI so that I can monitor the accomplishment of my goal
Scenario Outline: Permanent Poor BMI prediction performance
  Given the trained model
  And the previous BMI predicted
  When the difference between predicted BMI and read BMI is higher than $<threshold>
  And the number of recurring occurrences is higher than $<recurrence>
  Then the model is retrained
  Examples:
    | threshold |  | recurrence |
    |    0.2    |  |     3      |
Scenario Outline: Temporary Poor BMI prediction performance
  Given the trained model
  And the previous BMI predicted
  When the difference between predicted BMI and read BMI is higher than $<threshold>
  And the number of recurring occurrences is less than $<recurrence>
  Then the number of recurring occurrences is updated
  Examples:
    | threshold |  | recurrence |
    |    0.2    |  |     3      |
Scenario Outline: Acceptable BMI prediction performance
  Given trained model
  And the previous BMI predicted
  When the difference between predicted BMI and read BMI is smaller than $<threshold>
  Then the number of recurring occurrences is reseted.
  Examples:
    | threshold |
    |    0.2    |
```

Listing 1 – Acceptance

(being t+1 the prediction time) can positively influence the output (i.e., increases BMI). Plus, not only the weighted contribution of features on the output is difficult to predict, but they can change over time, whose phenomenon is the origin of the *concept drift* concept [20].

Concept drift refers to the phenomenon where the statistical properties of a target variable or the relationships between variables in a model change over time. In other words, it occurs when the underlying data distribution that a model was trained on no longer remains constant and evolves. A concept drift scenario includes features that describe food intake. These features are not listed in Fig. 3 because their contribution to the output is insignificant. One reason for that can be the stable food intake routine of people, which makes exercise the only variable that influences BMI. However, a new diet can make food intake features relevant, generating a concept drift.

Explainability helps to understand the nontrivial impact of each feature on the output. However, building acceptance criteria based on individual features is difficulted by the complex relationships between the input and output. But we can use the feature distribution of data to evaluate data completeness. We consider two metrics for the specification and validation of completeness:

- Data range represents the interval of values available for each numerical variable in the dataset. It can be represented by the minimum and maximum values or a lower and upper percentile to filter outliers.
- Data distribution determines the data distribution homogeneity of each variable. We represent that

```
Feature: Train model
User Story: As a Performance Monitor, I want to retrain models so that the minimum performance
thresholds can be ensured.
Scenario Outline: Model testing with low performance (appropriateness)
  Given a model trained with the training dataset
  And a testing dataset
  When the root mean square error of the testing dataset is higher than $<minvalue>
  Then the dataset is rejected
  Examples:
    | minvalue |
    |   0.5    |
Scenario Outline: Data range with a maximum value below threshold (completeness)
  Given a model trained with the training dataset
  When the maximum value of each $<feature> is lower than $<maxvalue>
  Then the model is rejected
  Examples:
    | feature           |    maxvalue    |
    | TotalDistance_t-1 |      5000      |
    | TotalSteps_t-2    |      10000     |
Scenario Outline: Data range with a minimum value below threshold (completeness)
  Given a model trained with the training dataset
  When the minimum value of each $<feature> is higher than $<minvalue>
  Then the model is rejected
  Examples:
    | feature           |    minvalue    |
    | TotalDistance_t-1 |       5        |
    | TotalSteps_t-2    |       5        |
Scenario Outline: Feature data not uniformly distributed (
  Given a model in compliance with the data range thresholds
  When the p-value of the chi-square statistic testing whether observed data follows a uniform
distribution, for each feature $<feature>, is lower than 0.05
  Then the dataset is rejected
  Examples:
    | feature           |
    | TotalDistance_t-1 |
    | TotalSteps_t-2    |
```

Listing 2- Impact of each feature on the model output.

distribution using the variance or standard deviation.

In Listing 2 the acceptance criteria specify thresholds against these metrics to evaluate data distribution homogeneousness over the amplitude of important features, determined from explainable models, such as that presented in Fig. 3.

6. Discussion

We challenged the assumption that conventional RE methods are unsuitable for ML needs. Instead, we showcased that it is feasible to articulate requirements using agile elements like user stories and acceptance criteria. The recent advent of interpretability methods for ML models has

made it achievable to specify requirements effectively.

Completeness is the most challenging data quality requirement since it depends on training data properties, which are more complex to specify than model performance. We utilized SHAP values to validate completeness, as they offer valuable feature boundaries that indicate their influence on the prediction outcome. The chi-square statistic is incorporated in the data acceptance criteria to determine whether the data of important features follow a uniform distribution. The rationale for that criterion is verifying if the values in the training dataset cover uniformly the amplitude of values of each important feature. That means that data quality requirements ensured the model exhibits satisfactory prediction performance for the test dataset and that all the relevant features have provided training values that cover their space uniformly. This requirement provides model resilience when it is used with production data.

7. Conclusion

Requirements engineering in machine learning projects presents wideband challenges. This paper cover specification of requirements for validating the model appropriateness and completeness of training datasets. We applied common agile artifacts (i.e., user stories and acceptance criteria) to the requirements specification of a personalized nutrition and exercise case study. It includes model and data requirements specification. We demonstrated that agile artifacts are adequate for specifying ML requirements.

Completeness criteria depend on data quality requirements that explainable models can support. We used SHAP to evaluate the uniformity of feature data in the training dataset to ensure coverage of feature space and thus provide guarantees that new production data have correspondence in the feature space of training data.

Concept drift is handled by requirements by ensuring model training when the prediction performance decreases below a threshold level.

Our work allows software engineers to specify ML requirements using agile methodologies, but it requires more validation using different project types. The ML problem space is wide and may require adaptation of requirements specification for some setups (e.g., those employing unstructured data as images). In future work, we plan to evaluate requirements specification in new scenarios, such as image classification. We also plan to study new approaches for describing data completeness using explainability methods.

Acknowledgements

National Funds fund this work through the FCT—Foundation for Science and Technology, I.P., within the scope of the project Ref. UIDB/05583/2020. Furthermore, we thank the Research Centre in Digital Services (CISeD) and the Polytechnic of Viseu for their support.

Author contributions

Carlos Cunha: Conceptualization, Methodology **Rui Duarte:** Software Validation. **Rafael Oliveira:** Field Study.

Conflicts of interest

The authors declare no conflicts of interest.

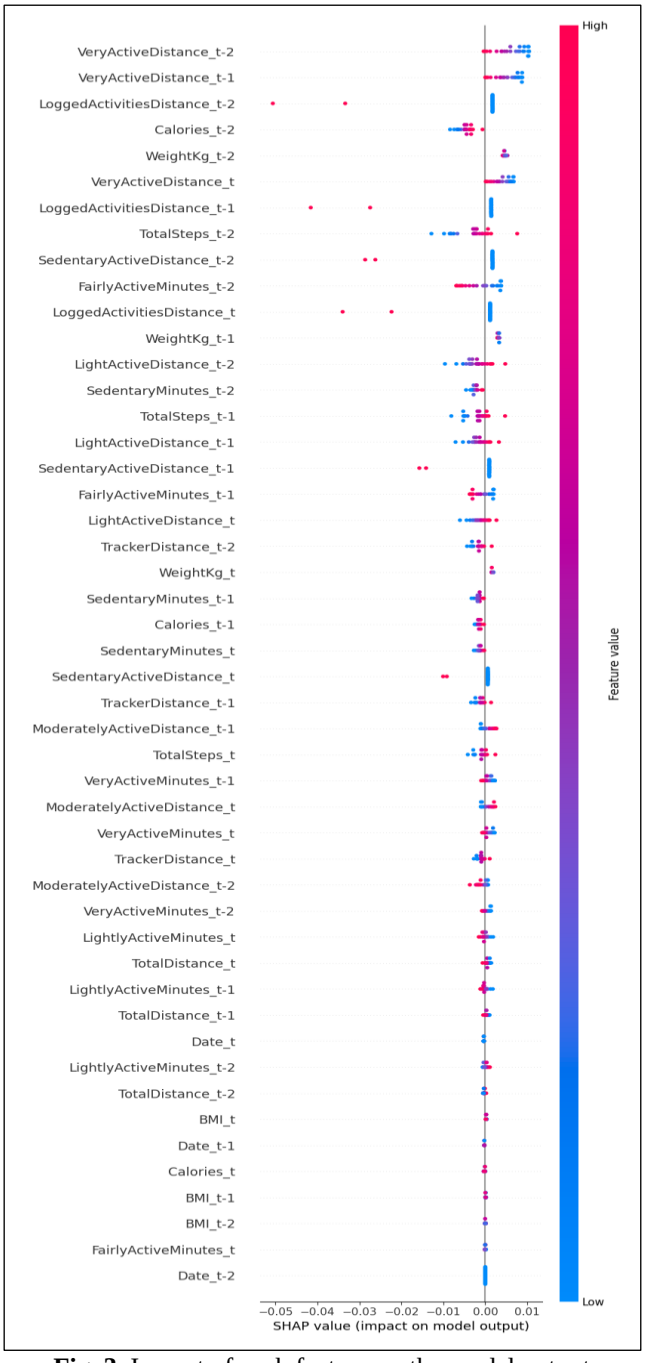

Fig. 3. Impact of each feature on the model output

References

[1] A. Vogelsang e M. Borg, «Requirements Engineering for Machine Learning: Perspectives from Data Scientists», em 2019 IEEE 27th International Requirements Engineering Conference Workshops (REW), set. 2019, pp. 245–251. doi: 10.1109/REW.2019.00050.

[2] M. Cohn, User stories applied: for agile software development, 18. print. em Addison-Wesley signature series. Boston, Mass.: Addison-Wesley, 2013.

[3] K. Ahmad, M. Abdelrazek, C. Arora, M. Bano, e J. Grundy, «Requirements engineering for artificial intelligence systems: A systematic mapping study», Inf. Softw. Technol., vol. 158, p. 107176, jun. 2023, doi: 10.1016/j.infsof.2023.107176.

[4] F. Ishikawa e Y. Matsuno, «Evidence-driven Requirements Engineering for Uncertainty of Machine Learning-based Systems», em 2020 IEEE 28th International Requirements Engineering Conference (RE), ago. 2020, pp. 346–351. doi: 10.1109/RE48521.2020.00046.

[5] X. Wang, «A framework for Requirements specification of machine-learning systems», apresentado na The 34th International Conference on Software Engineering and Knowledge Engineering, jul. 2022, pp. 7–12. doi: 10.18293/SEKE2022-143.

[6] Z. Pei, L. Liu, C. Wang, e J. Wang, «Requirements Engineering for Machine Learning: A Review and Reflection», em 2022 IEEE 30th International Requirements Engineering Conference Workshops (REW), Melbourne, Australia: IEEE, ago. 2022, pp. 166–175. doi: 10.1109/REW56159.2022.00039.

[7] K. M. Habibullah e J. Horkoff, «Non-functional Requirements for Machine Learning: Understanding Current Use and Challenges in Industry», em 2021 IEEE 29th International Requirements Engineering Conference (RE), Notre Dame, IN, USA: IEEE, set. 2021, pp. 13–23. doi: 10.1109/RE51729.2021.00009.

[8] S. Nalchigar, E. Yu, e K. Keshavjee, «Modeling machine learning requirements from three perspectives: a case report from the healthcare domain», Requir. Eng., vol. 26, n.º 2, pp. 237–254, jun. 2021, doi: 10.1007/s00766-020-00343-z.

[9] S. Nalchigar, E. Yu, e R. Ramani, «A Conceptual Modeling Framework for Business Analytics», em Conceptual Modeling, I. Comyn-Wattiau, K. Tanaka, I.-Y. Song, S. Yamamoto, e M. Saeki, Eds., em Lecture Notes in Computer Science, vol. 9974. Cham: Springer International Publishing, 2016, pp. 35–49. doi: 10.1007/978-3-319-46397-1_3.

[10] H. Challa, N. Niu, e R. Johnson, «Faulty Requirements Made Valuable: On the Role of Data Quality in Deep Learning», em 2020 IEEE Seventh International Workshop on Artificial Intelligence for Requirements Engineering (AIRE), Zurich, Switzerland: IEEE, set. 2020, pp. 61–69. doi: 10.1109/AIRE51212.2020.00016.

[11] H. Kuwajima, H. Yasuoka, e T. Nakae, «Engineering problems in machine learning systems», Mach. Learn., vol. 109, n.º 5, pp. 1103–1126, mai. 2020, doi: 10.1007/s10994-020-05872-w.

[12] F. Febrero, C. Calero, e M. Ángeles Moraga, «Software reliability modeling based on ISO/IEC SQuaRE», Inf. Softw. Technol., vol. 70, pp. 18–29, fev. 2016, doi: 10.1016/j.infsof.2015.09.006.

[13] B. D. Klein, «User Perceptions of Data Quality: Internet and Traditional Text Sources», J. Comput. Inf. Syst., vol. 41, n.º 4, pp. 9–15, 2001, doi: 10.1080/08874417.2001.11647016.

[14] S. Hochreiter e J. Schmidhuber, «Long Short-Term Memory», Neural Comput., vol. 9, n.º 8, pp. 1735–1780, nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[15] E. Layer, «Fitbit Tracker Data Analysis». https://www.kaggle.com/code/ericlayer/fitbit-fitness-tracker-data-analysis.

[16] S. M. Lundberg e S.-I. Lee, «A Unified Approach to Interpreting Model Predictions», em Advances in Neural Information Processing Systems, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, e R. Garnett, Eds., Curran Associates, Inc., 2017. [Em linha]. Disponível em: https://proceedings.neurips.cc/paper_files/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf

[17] R. Alenezi e S. A. Ludwig, «Explainability of Cybersecurity Threats Data Using SHAP», em 2021 IEEE Symposium Series on Computational Intelligence (SSCI), Orlando, FL, USA: IEEE, dez. 2021, pp. 01–10. doi: 10.1109/SSCI50451.2021.9659888.

[18] S. Ahmed, S. N. Nobel, e O. Ullah, «An Effective Deep CNN Model for Multiclass Brain Tumor Detection Using MRI Images and SHAP Explainability», em 2023 International Conference on Electrical, Computer and Communication Engineering (ECCE), Chittagong, Bangladesh: IEEE, fev. 2023, pp. 1–6. doi: 10.1109/ECCE57851.2023.10101503.

[19] M. Yap et al., «Verifying explainability of a deep learning tissue classifier trained on RNA-seq data»,

Sci. Rep., vol. 11, n.º 1, p. 2641, jan. 2021, doi: 10.1038/s41598-021-81773-9.

[20] G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, e F. Petitjean, «Characterizing concept drift», Data Min. Knowl. Discov., vol. 30, n.º 4, pp. 964–994, jul. 2016, doi: 10.1007/s10618-015-0448-4.