Use case instructions

Requirements

Hardware: Bring your own device.
Connectivity: An internet connection is necessary to access the dataset and test it.
Domain Knowledge: A little knowledge about diabetes can be useful.
Account on Kaggle: To submit the result of the hackathon.
Software: See the table with the different options at the end of the document.

Dataset

Description

The training dataset comprises 3000 entries, while the test endpoint contains 300 entries. This dataset has been adapted from an original source for the purpose of this challenge.

Access

You can explore the API using the OpenAPI spec at `/openapi.json`, you can paste it into Swagger UI or Postman to get interactive docs.

For example:

- Swagger Editor → Paste your /openapi.json → instant documentation site.
- Postman → "Import → Link" → paste your spec URL.

Base URL: `https://api.hackupm2025.workers.dev`

Getting Started:

Linux/MacOS

- Health Check: `curl "https://api.hackupm2025.workers.dev/health"`
- API Documentation: `curl "https://api.hackupm2025.workers.dev/openapi.json"`

Windows:

- Health Check: `Invoke-WebRequest "https://api.hackupm2025.workers.dev/health" | Select-Object -Expand Content`
- API Documentation: `Invoke-WebRequest "https://api.hackupm2025.workers.dev/openapi.json"`

Notes:

- The API limits responses to a maximum of 100 objects per request.

Evaluation

The total score of each team will be a combination of three key areas:

The total score for each team will be the result of combining four key areas: **Technical Performance**, **Process Quality**, **Innovation and Utility, and Communication**.

1. Model Technical Performance (Weight: 40%)

This is the most objective pillar of the evaluation.

- **Main Metric: F1 Score**. This measures the harmonic mean of **Precision** (avoiding false positives) and **Recall** (avoiding false negatives). A high F1 Score indicates a well-balanced model.
- **Evaluation with hidden dataset:** The final F1 Score for each team will be calculated **exclusively** on a test set that participants have not seen (the *hidden test set*).
- **Procedure:** Teams will be ranked according to their F1 Score obtained on this hidden dataset.

2. Process Quality (Weight: 30%)

This evaluates the team's methodology, rigor, and independence.

- **Feature Engineering:** Evaluation of creativity in handling and transforming variables to improve prediction.
- **Modeling:** Selection of the appropriate classification algorithm and optimization of its hyperparameters.
- **Code and Reproducibility:** The code must be clean, documented, and easy to execute.
- **Autonomy:** The team's ability to solve problems independently will be evaluated.

Received Assistance	Criterion Score	Description
None	**Maximum**	The team solved all technical challenges by themselves.
Low	**Good**	Specific, clear, and highly specific queries (e.g., a minor platform error).
Medium	**Regular**	Requests for help with *debugging* complex code errors, or questions about algorithm selection.
High	**Minimum**	Need for frequent assistance, help with basic programming problems, or requiring detailed guidance to advance in the process.

3. Innovation and Utility (Weight: 20% - Max. 20 points)

This evaluates the ability to present the solution and generate value.

- **Innovation:** Novelty or creativity in the application of classification techniques or in the data preprocessing approach.
- **Utility and Actionability** (demo): This criterion assesses how well the team transformed the binary predictive model into an usable application by providing clear, context-specific evidence for its prediction, using the clinical notes.

4. Presentation (Weight: 10%)

- **Clarity of Presentation:** Fluid and concise explanation of the problem, process, and results, including the interpretation of the F1 Score.
- **Conclusions and *Insights***:** Identification of the most important variables (*feature importance*) and the translation of metrics into **practical recommendations** for the audience or business.

1. Model Technical Performance (Weight: 40% - Max. 40 points)

This criterion is **objective** and is calculated based on the F1 Score obtained on the hidden test set.

Criterion	Weight	Evaluation Scale	Score
F1 Score	**40%**	**Ranking** (by position)	Teams are ranked strictly according to their F1 Score.
		1st Place	**40 points**
		2nd Place	**35 points**
		3rd Place	**30 points**
		4th Place onwards, gradually decreasing.	

2. Technical Quality and Process (Weight: 30% - Max. 30 points)

The Machine Learning methodology, code, and team autonomy are evaluated. (Scale 1-4 for each criterion. All 4 criteria contribute equally to 30% of the total weight).

Criterion	Description
Feature Engineering	**4 (Outstanding):** Complex and well-founded transformations, excellent handling of null and outlier data. **1 (Needs Improvement):** Features were used only as provided, or null/outlier values were handled incorrectly.
Modelling and Tuning	**4 (Outstanding):** Appropriate algorithm selection and hyperparameter optimization. **1 (Needs Improvement):** Use of a simple model without any hyperparameter optimization.
Code and reproducibility	**4 (Outstanding):** Clean, well-commented, and structured code, using a reproducible environment (e.g., requirements or dependency file).
Autonomy	Scale 1–4 defined in the following table.

Level of Assistance Received	Criterion Score (1-4)
None	4
Low	3
Medium	2
High	1

3. Innovation and Communication (Weight: 30% - Max. 30 points)

The team's ability to explain and generate value is evaluated. (Scale 1-4 for each criterion. All 3 criteria contribute equally to 30% of the total weight).

Criterion	Description
Innovation and Creativity	**4 (Outstanding):** Application of an ingenious approach or creative use of tools. **1 (Needs Improvement):** Use of a predictable and standard

	approach.
Clarity of Presentation	**4 (Outstanding):** The solution is communicated concisely, clearly, and with effective visualizations within the allotted time. **1 (Needs Improvement):** The explanation was confusing, time was exceeded, or the logic of the model was not understood.
Conclusions and Insights	**4 (Outstanding):** The team translates the F1 Score and identifies key variables (feature importance) to generate practical recommendations. **1 (Needs Improvement):** Only the final metric is presented without an explanation of what drives the prediction or what it means for the business problem.

FINAL SCORE CALCULATION (Out of 100 points)

The final score is calculated by mapping the 1-4 scale score to the assigned percentage weight.

1. **F1 Score (P40):** The score is taken directly from the ranking table (Max. 40).

2. **Quality Score (P30):** The **4** scores from section 2 are summed and scaled to 30. (Maximum total points is 4 X 4 = 16).

$$Quality = \left(\frac{\sum (Quality\ scores\ 1{-}4)}{16} \right) \times 30$$

3. **Communication & Innovation Score (P30):** The **3** scores from section 3 are summed and scaled to 30. (Maximum total points is 3 X 4 = 12).

$$Com\&Innovation = \left(\frac{\sum (Innovation\ scores\ \ 1{-}4)}{12} \right) \times 30$$

Total Score = F1 Score + Quality Score + Com&Innovation Score

Tools

Basic Machine Learning Environments and Libraries			
Category	*Description*	*Open Source Tool*	*Commercial Tool*
Data Manipulation / Cleaning	**Data ingestion (from API)**, feature creation, and data type transformation (for tabular data).	**Pandas** (Link)/ **dplyr** (Link) / **data.table** (Link)	**Microsoft Excel / SQL (Enterprise)** (Excel Link)
ML Algorithms	Establish a robust and interpretable **baseline classification model** (e.g., Logistic Regression, Decision Trees).	**Scikit-learn** (Link) / R (Link)	**SAS Viya** (Link)
High-Performance ML	Achieve **maximum predictive accuracy** on structured data using advanced *ensemble* methods.	**XGBoost / LightGBM** (XGBoost Link)	**MATLAB Machine Learning Toolbox** (Link) [1]
Natural Language Processing (NLP)	**Efficient basic text processing**, such as Named Entity Recognition (NER), to extract simple structured variables from clinical notes.	**NLTK (Link) / SpaCy (Link) / tm** (Link) / **Hugging Face Ecosystem** (Link)	**Google Cloud Natural Language API / IBM Watson Discovery (Google NLP Link)**
Statistical Analysis	Rigorous **hypothesis** testing, calculation of confidence intervals, and **statistical explainability** of features.	**Statsmodels** (Link) /broom (link)	**IBM SPSS Statistics / Stata** (SPSS Link / Stata Link) [1]
Platforms	**End-to-end environment** for data preparation, model development, and deployment.	**KNIME, H2O.ai**	**Dataiku, Databricks**
Visualization	Presentation of results in **graphical format**.	**Matplotlib, gradio, streamlit, grafana, ggplot2**	**Plotly** (link), **Tableau**

Generative Models and Tools			
Category	*Description*	*Open Source Tool*	*Commercial Tool*
Large Language Models	Feature Extraction (structuring text into variables), Embedding Generation, or Direct Classification through prompting or fine-tuning..	**Raw Models** **LLaMA 3** (Link), **Mistral / Mixtral** (Link), **Gema** (Link) **Model Repository** **Hugging Face** (Link) **Cloud Execution Platforms** **Kaggle** (Link), **Google Colab** (Link) **Local Execution Platforms** -> **Ollama** (Link), **LM Studio** (Link), **GPT4All** (Link) **Free APIs** **Hugging Face** (Link)	**Free Options** **Google AI Studio** (Link, modelo 'gemini-2.5-flash'), **Groq** (Link, 'Developers -> free API key', variedad de modelos) [2] **Paid Options** **OpenAI** (Link) , **Google - Gemini** (Link) , **Claude** (Link), **Cohere** (Link)
Core Frameworks	Construction and training of custom Deep Learning models, or serving as the backend (support) for the Fine-Tuning of LLMs.	**PyTorch** (Link) / **TensorFlow** (Link) / **Keras**	**MATLAB Deep Learning Toolbox** (Link) [1]
Transformers	Access, load, and tokenize pre-trained transformer models (e.g., BERT) to generate text features (embeddings).	**Hugging Face Ecosystem** (Link)	**Amazon SageMaker JumpStart / Google Vertex AI Model Garden** (Vertex AI Link)

IDEs and Development Environments			
Category	*Description*	*Open Source Tool*	*Commercial Tool*
Integrated Development Environment (IDE)	Write, debug, and manage Python/R code locally.	**VS Code** (Core is OS) (Link)/ **Positron** (Link)	**PyCharm Professional** (Link) [2]
Notebooks	Data exploration, visualization, and rapid prototyping by combining code and narrative.	**Jupyter Notebooks / Lab** (Link) **/ markdown / quarto** (Link)	**Google Colab (Link) / Databricks Notebooks (Link)**
Version Control	For collaboration, version control, and a reproducible workflow.	**Git** (link) **/ Codeberg** (Link)	**GitLab (link) / Azure DevOps (Link) / GitHub (Link)**

[1] UPM has licenses for these products.

[2] One-year free trials for university students in Spain.