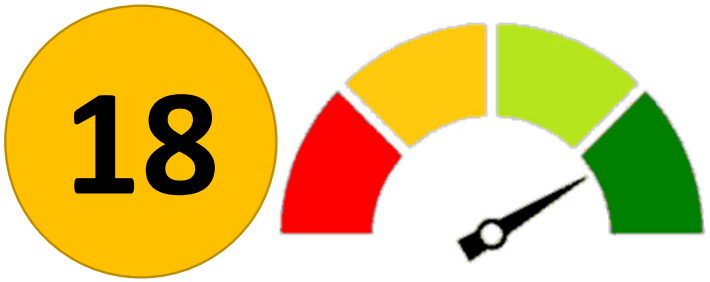
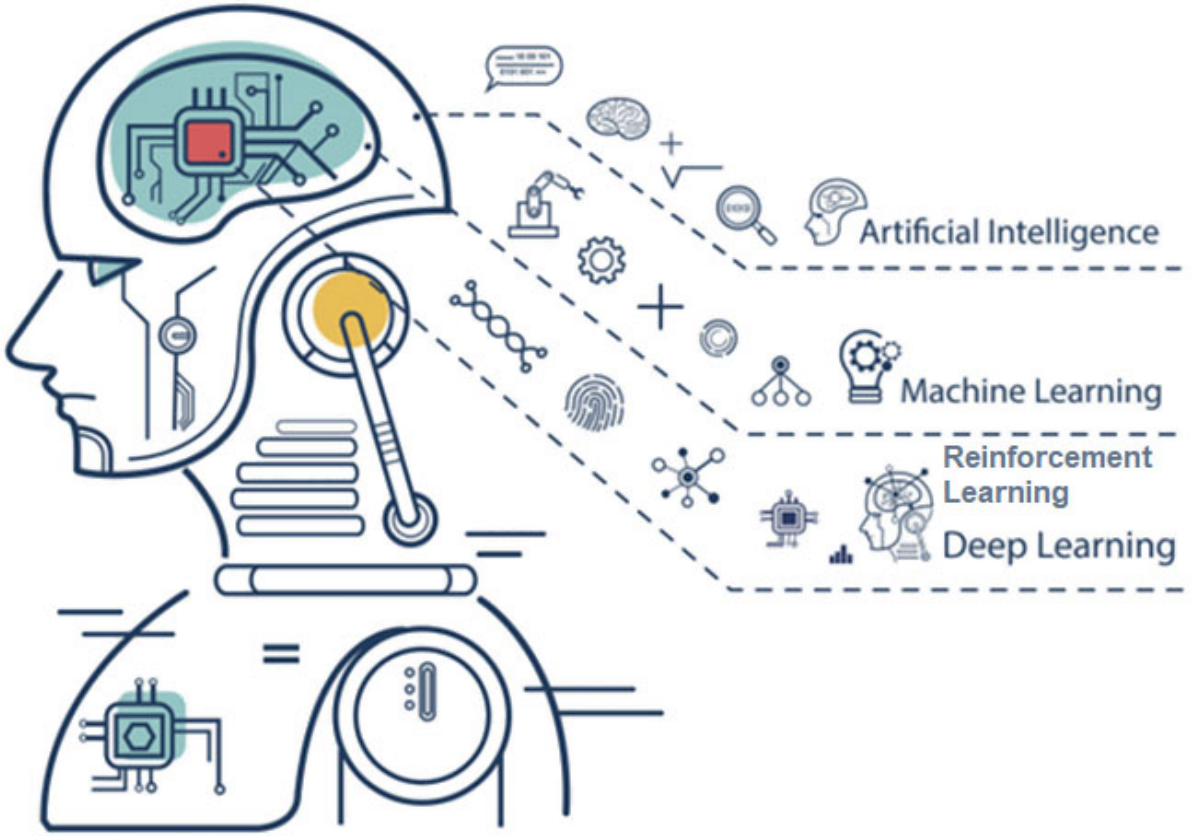
Designing Artificial Intelligence & Implementing Smart Technologies

DAI



José Manuel Rey, IE



Designing Artificial Intelligence & Implementing Smart Technologies

Group Assignment 18.00

Date: Oct/30 2025

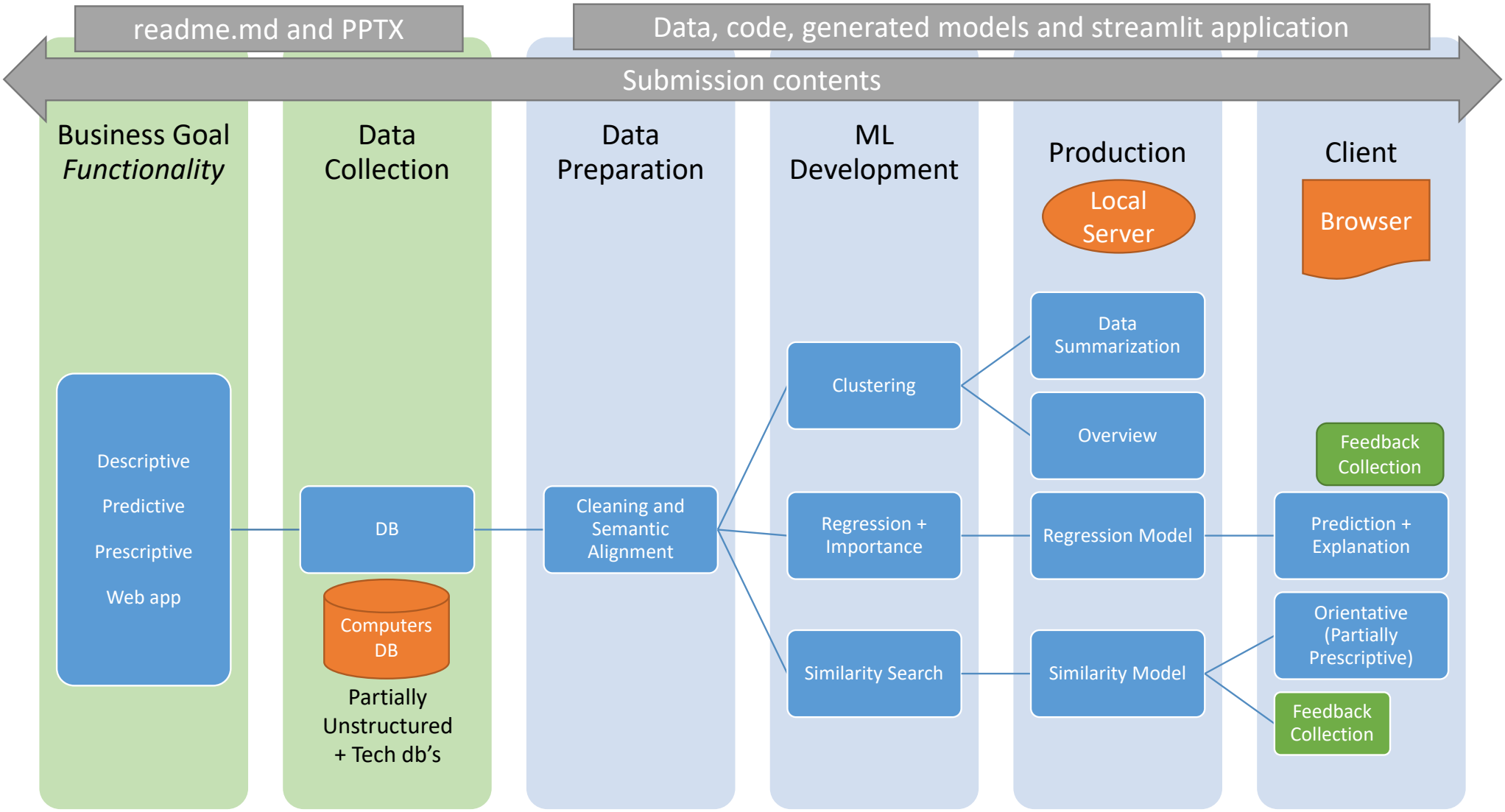
Due Date: Dec/9 2025 [before SESSION 29]

Ongoing grading (maximum)

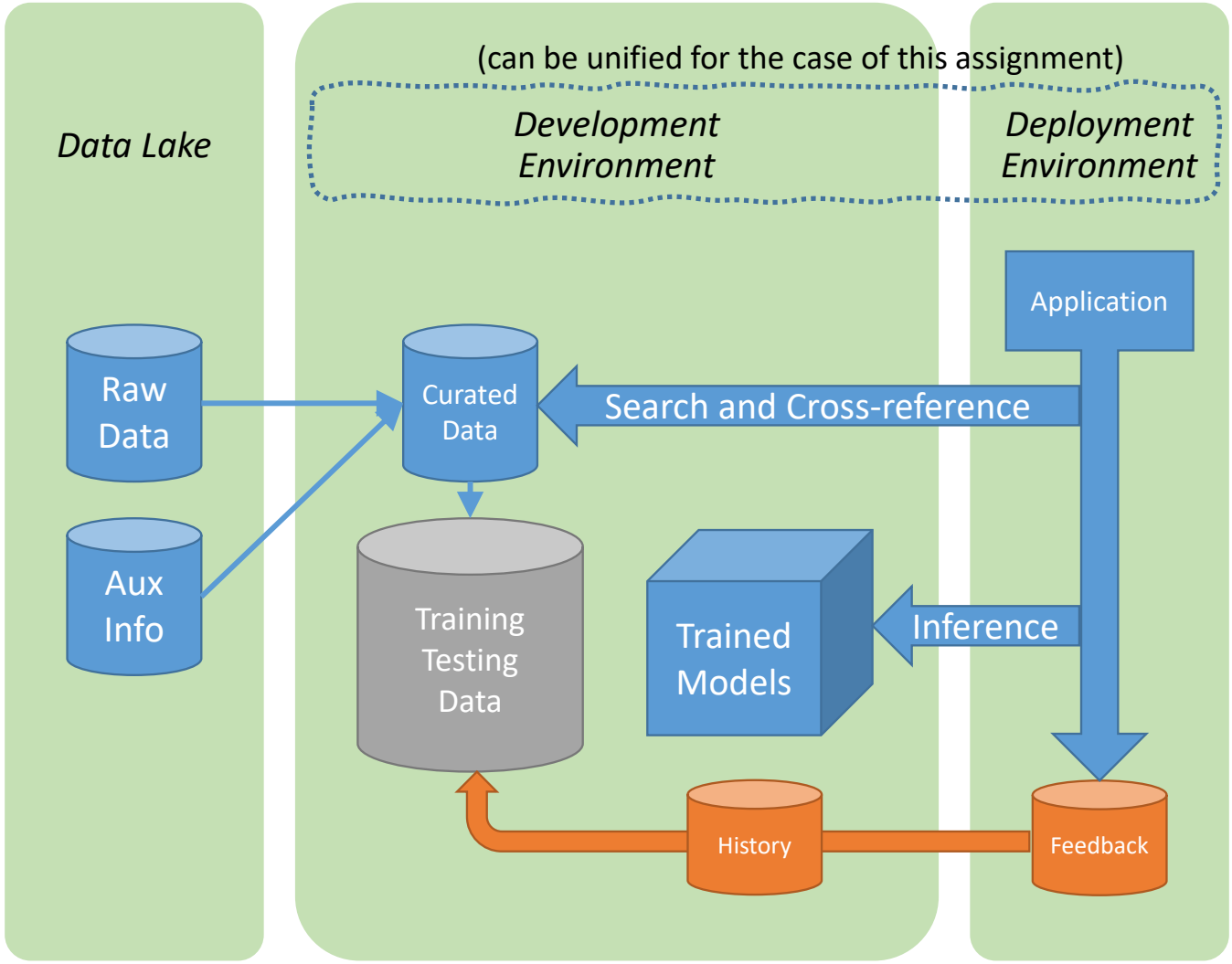
180 Group Assignment

Assignment Description Overview

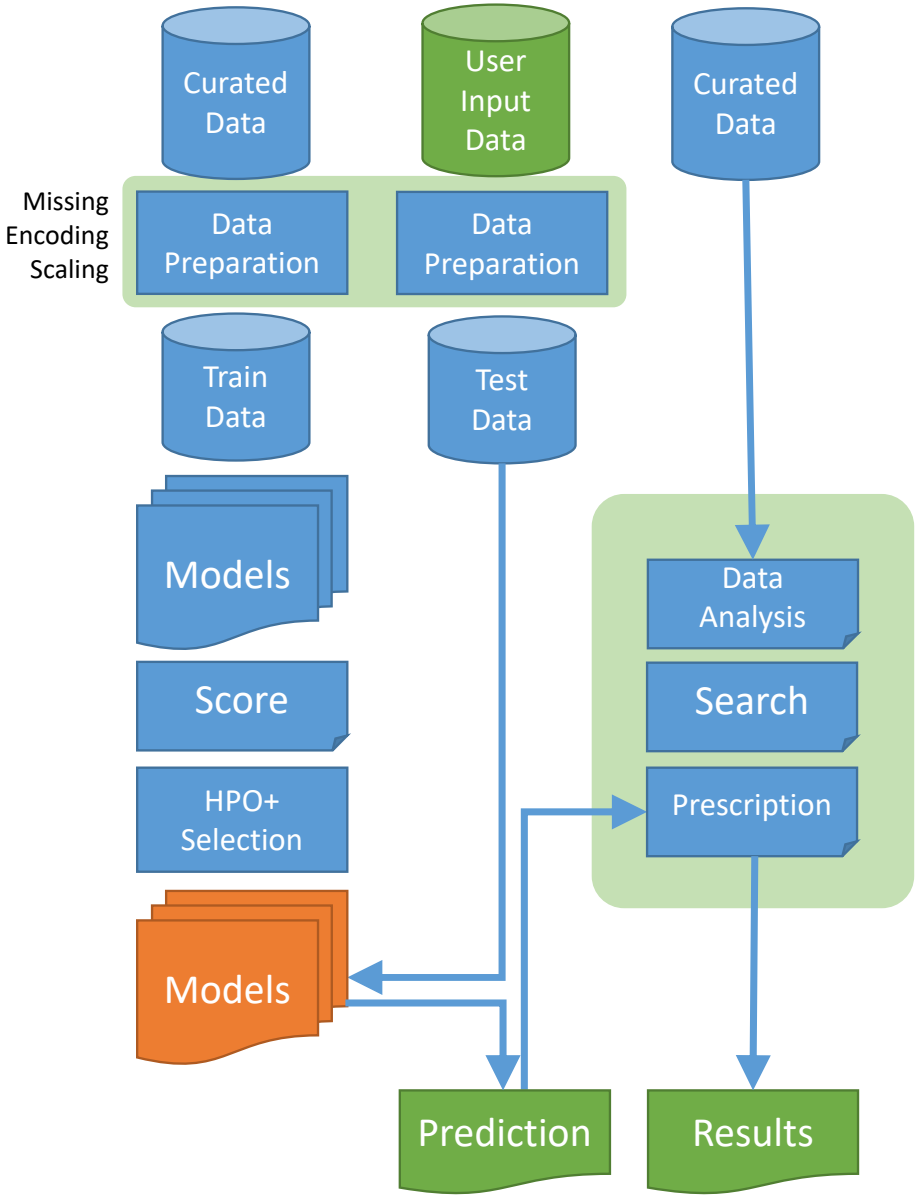
- The objective of the Assignment is developing a working application (start-to-end) integrating a ML workflow approach
- It will comprise the different stages covered during the course:
  - Business Goals and Functional/Task analysis
  - Data Analysis and Preparation
  - Feature Engineering and Feature Selection
  - Algorithm selection, model training and validation and model selection
  - Expected performance metrics estimation
  - Model deployment and Model serving
  - User Interface (NLP + explainability + feedback)


readme.md and PPTX
Data, code, generated models and streamlit application
Submission contents
Business Goal
Functionality
Data Collection
Data Preparation
ML Development
Production
Client
Local Server
Browser
Descriptive
Predictive
Prescriptive
Web app
DB
Computers DB
Partially Unstructured + Tech db's
Cleaning and Semantic Alignment
Clustering
Regression + Importance
Similarity Search
Data Summarization
Overview
Regression Model
Similarity Model
Feedback Collection
Prediction + Explanation
Orientative (Partially Prescriptive)
Feedback Collection

Database Management



Typical Application Flow



Note: coding

- Try to keep names of original fields unchanged (for better code maintainability and traceability)
- Use starting underscore for naming new features created
- Code organization:
  - Jupyter notebooks for the main “sections” of the ML workflow (data preparation, model selection and training, model validation, …) [sequentially numbered]
  - Python code for the common/shared libraries or modules created and for the streamlit frontend (main and pages)

The case of prediction with “incomplete” information (missing fields)

A) Use specific sub-model trained with the partial set of features required

- In the case of RandomForest or other similar models you can restrict to the “sub-trees” trained with that subset (if any), but typically you must have it prepared (trained) in advance

B) Treat as “missing” and do the standard missing imputations defined (if applicable)

C) Generate multiple values (sample) for the missing fields, within the allowed ranges, generate multiple predictions, and produce a combined result

Use the “feature importances” generated by the trained models to offer an orientation about the relevant input fields required for the predictions. Organize the input fields by blocks ranked by importance and offer the “user” the possibility of only informing the blocks or highest relevance [This will typically required having a set of pre-trained models for each combination of blocks, with some fields as “mandatory”, and then applying some of the above techniques for the “missing” values in the blocks]

Basic work program

- Coordination/Planning and Project scheduling
- EDA, Data Preparation and Feature Engineering/Selection
- Model(s) Training/Testing/Validation
  - Model Selection → Hyperparameter tuning
  - Model Card: description, expected performance, explainability issues
- Frontend functionality
  - Standard data-oriented interfaces
  - NLP input and output presentation “agents”
  - Streamlit development
- Testing/Debugging: Units/Integration
- Presentation and report

Assignment Description Overview

- For the development of the project, the students may use any of the tools used most often during the course (python, pandas, numpy, sklearn, pytorch, tensorflow, matplotlib, seaborn, huggingface, streamlit, etc)
- However, if for any reason, they find convenient to implement the solution with any other tools, that could be equally valid, provided that they include the necessary information for the correct deployment and automatic execution on any standard deployment server (from either a simple *requirements.txt* to a full *docker image and dockerfile* if necessary)
- As for the algorithms to be used to match the functionalities required, the solution can be effectively developed just by using some of the essential ones exemplified during the course (knn, kmeans, PCA, SVM, decision trees, various regressors and classifiers, various ensemble methods, MLP or NN, etc as well as an standard “small” open-source LLM agent for the NLP interfaces).
- Again, if for any reason, should any other type of algorithms be deemed as more suitable for any of the tasks involved (cosine similarity, latent semantic analysis, text embeddings, hierarchical density clustering, etc..) its use would also be allowed if correctly documented the reasons for such decision (either conceptual or pure practical/empirical based on the resulting metrics achieved)
- The use of automatic/generative coding (or documenting) tools is totally valid, with just the proper specific acknowledgment of their utilization and the documentation of the prompts used.

Data Collection

To understand the main structure and issues of the dataset, take a first look to the file: *db_computers_columns_names.txt*

- You are provided with a Database of computer products in a marketplace
- Partially unstructured (as it could be typically found in a corporate data-lake)
  - 8064 rows × 135 columns, Data encoding is 'utf-8-sig'
  - Redundant fields and "unclassified" information
  - The features and their content are in the marketplace language (Spanish) and the monetary and measurement units are in the European Metric System (except some particular cases)
  - Most of the numerical data has descriptors or added texts that have to be properly removed (cleaning)
  - There is plenty of missing data (specially for the "granular" fields, optional or very specific features or characteristics, and so) and, in many cases, it is a "Missing-Not-At-Random" scenario (as it is conditioned on some other feature/type/class of the dataset)
  - In other cases, data could be equivalent but formulated in a different form:
    - Example: "Pantalla_Tamaño de la pantalla" and "Pantalla_Diagonal de la pantalla", both describe the same characteristic but with different measurement criteria
  - Categorical features will be (in general) of type "multilabel" (separated by commas) with the implicit assumption that the **first** "label" is the most relevant in terms of the characterization of the feature

Data Collection

For the proper first load of the provided database just use:

```
import pandas as pd
df = pd.read_csv( 'db_computers_2025_raw.csv’,
                  encoding='utf-8-sig’,
                  index_col=0,
                  low_memory=False
                ).reset_index(drop=True)
```

Auxiliary Technical Databases

- For performing the encoding of some of the key “categorical” features that you may encounter (e.g: “Procesador_Procesador”, “Gráfica_Tarjeta gráfica”) do not perform any type of target encoding.
- Use instead the two enclosed datafiles
  - `db_cpu_raw.csv`
    - Fields: CPU Mark (higher is better) or Rank (lower is better)
  - `db_gpu_raw.csv`
    - Fields: Passmark G3D Mark (higher is better) or Rank (lower is better)
- Note: pairing the different fields will require some type of similarity search or fuzzy search comparison

Data Preparation
Additional observations

- The process of data collection/preparation involves necessarily, as you already know, the participation of Subject Matter Experts (SME) for the correct conceptual analysis of the data and its potential value.
- For this assignment, we will assume that you are qualified enough for acting as SME in this domain so your qualitative/conceptual assessments (for example when grouping types of resolution for the screens, processor GHz, or quality of the different WiFi protocols supported, etc) will be considered authoritatively correct
- One of the main roles of SME's at these stages is the pre-selection of the core set of common features relevant for further modelling → something equivalent to a greatest common divisor (GCD) of usable features
- This can be specially relevant in this case, as the sparsity of the dataset is huge in some of the "sub-features", heterogeneous depending on the basic type (laptops or desktops) and some aggregation could be ideal for using is potential informative value at its best

Data Preparation / Features Issues

As examples of the issues around features that you can expect finding in any basic real world implementation you can look at:

- Numerical features unnaturally splitted (potentially, or perhaps not…):
  - Example:
    - Capacidad disco duro (TB) > Hard drive (TB)
    - Capacidad disco duro (GB) > Hard drive (GB)
- Categorical combined (hot-encoded) features:
  - Example:
    - Bahía interna de 2,5" > Internal Bay of 2.5 "
    - Bahía interna de 3,5" > Internal Bay of 3.5 "
    - Bahía externa de 3,5" > External Bay 3.5 “
    - Bahía externa de 5,25" > External Bay 5.25 "

Proper feature engineering can help making this information more explicit and useful for the models

General Overview of Features

Type of Info	Group of features	(English translation)
Description	Título	Name
	Serie	Series
	Tipo	Type
	Tipo de producto	Subtype
	Adecuado para	Suggested use
Technical/Hard	Procesador	Processor
	Almacenamiento	Storage
	Disco duro	Hard disk
	RAM	RAM
	Conectividad	Connectivity
	Comunicaciones	Communications
Technical/Soft	Sistema operativo	Operating system
	IA	AI
HID	Gráfica	Graphic card
	Pantalla	Display
	Teclado	Keyboard
	Cámara	Camera
	Sonido	Sound
Physical	Propiedades de la carcasa	Casing
	Medidas y peso	Dimensions and weight
	Alimentación	Power supply
Other	Características especiales	Special features
	Otras características	Other features
Market	Precio (Rango)	Price (Range)
	Ofertas (Número)	Offers (Number)

Business Goal
Functional Analysis

We are requested to create a multi-page web application (browser client) that provides the following set of basic functionalities

- 1. Descriptive
  - (A) Statistical Overview/Summary of the market offerings
  - (B) Clustering/Segmentation of the different types of products
- 2. Predictive
  - 2.1 Prediction of the price of a particular computer configuration or model
    - Identify basic features
    - Provide “default” values for some of the features not informed explicitly by the client
      - Numerical values (either direct input or sliders)
      - Categorical (combo dropdown)
      - Yes/No features (checkboxes)
    - Request input for desired/necessary information
  - 2.2 Explanatory breakdown of the price on the basis (importance) of the main components or features
- 3. Prescriptive (orientative, partially prescriptive, similarity search)
  - Show the k-best offers similar to the requested features and map its distances in terms of characteristics, predicted price and real price

1

2

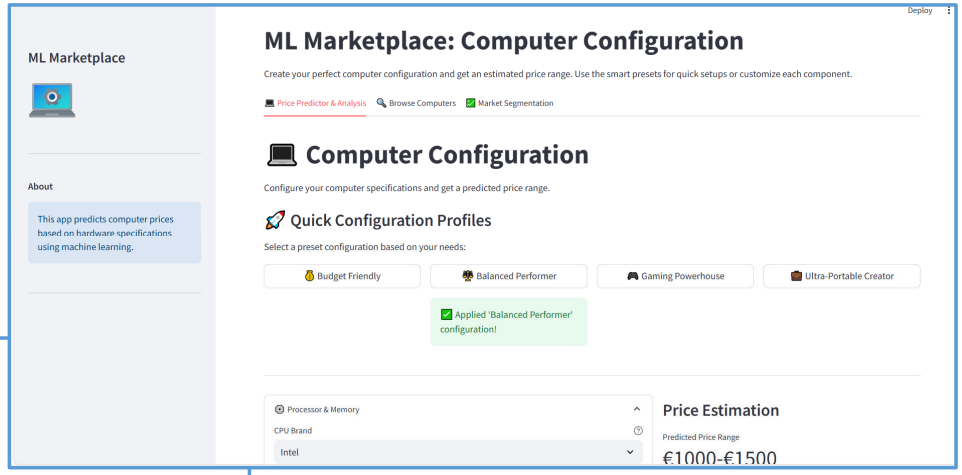
3

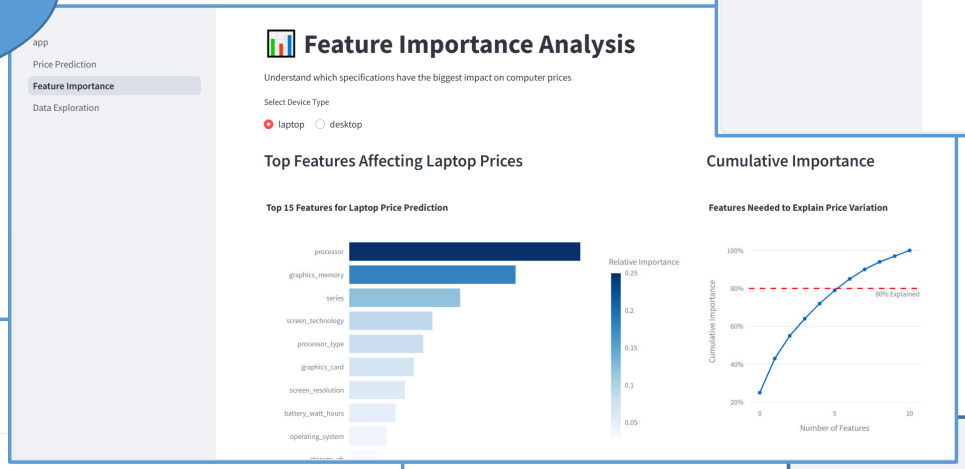
4

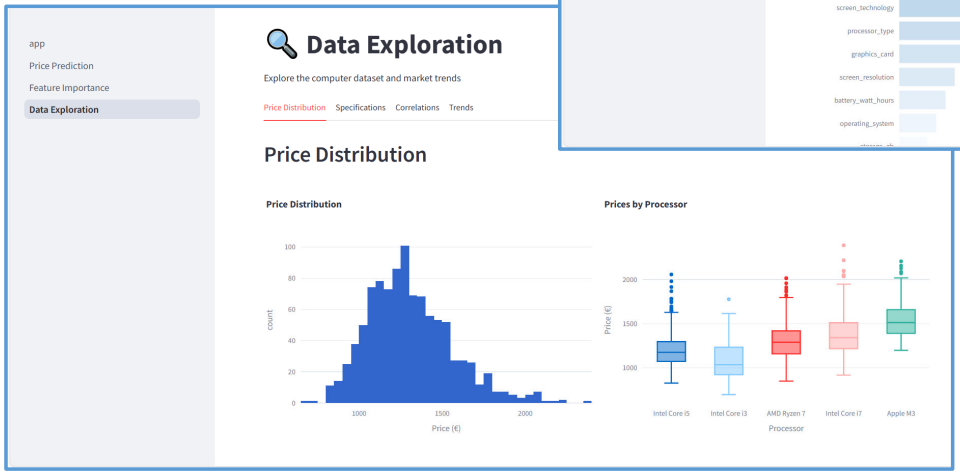
5 There could be an optional additional functionality chosen by the group

Note: UI

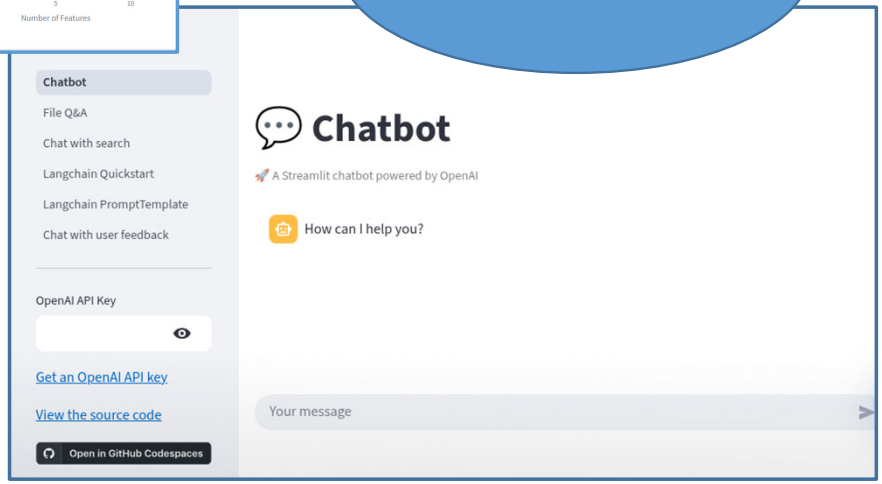
Traditional key input fields





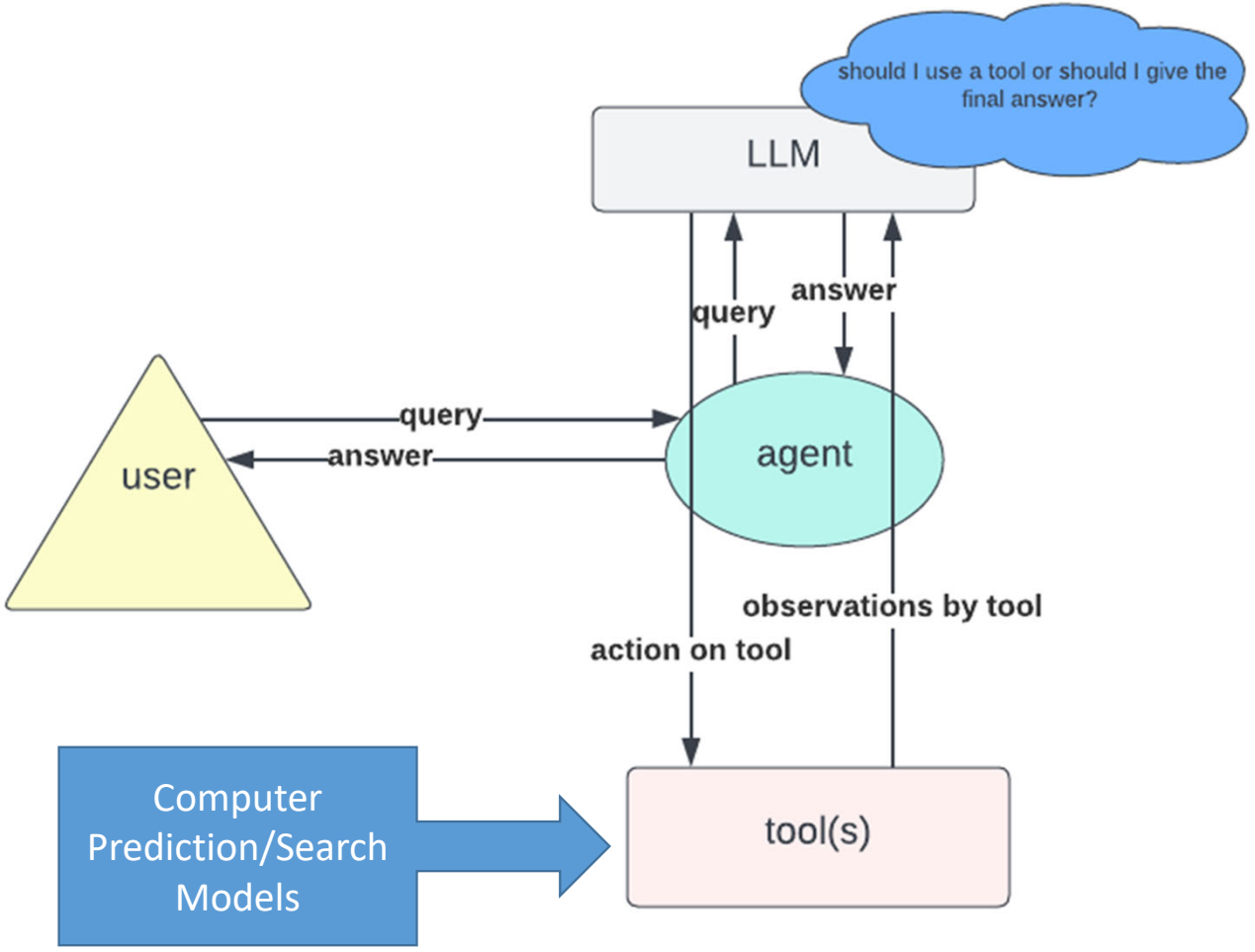


NLP



Note: UI

LLM interface could just be built using a feature extraction tool from the user query (e.g. LangExtract) or implemented as an standard function-calling [a.k.a tool-calling] scheme



Note: UI

- The language for the interface must be English → No need for a multi-language application development
- However, for the input elements related to the data-fields (typically lists or combo-boxes) the contents displayed to the user should be in the original dataset language (Spanish)
- No need for a translation of the dataset contents (either column names or values) is required. So, do not worry about using "*googletrans*" or any other library for a quick and dirty language conversion. Mixing languages in the visual interface items is totally acceptable.
- The dataset contains a total of ≃140 Spanish words (excluding "named entities") which seem to be directly easily interpretable. In case of doubt (for the semantic data preparation) use sparsely any translator tool or consult to any other of the team members
- Additionally, no UI tiptools or on-hover informative text is required, as neither it is any type of online help implementation. This is not any of the objectives of the assignment.
- Every functionality provided must include a (very simple) feedback evaluation option for the user that has to be recorded and stored to disk (basic flat file). This is an essential tool for the on-going monitoring of the application and the detection of quality issues and in some cases for the progressive extension of the training dataset.
- Keep everything as simple and straight-to-the-point as possible (just the conceptual core)

Typical user questions/inputs to be addressed

Data input:

- Predicted price for a laptop screen:15.6“, Intel Core i9-13900H RAM:32GB SSD:1TB, RTX 5060
- Best offers for desktop PC, i5, RTX 5060 SSD 1TB RAM 32GB
- Range of prices for Mini-PC, NVIDIA RTX 3090.

NLP input:

- Comparison of best buy opportunities (similarity metric vs predicted price vs real price) of desktops and mini-pc with Core Ultra 7 and NVIDIA graphic cards

Teamwork Organization

- Given the nature of the Assignment/Project, that requires the integration of multiple capabilities/processes, that span from
  - data preparation,
  - several different models training, refinement, validation and testing
  - functional integration of those components,
  - basic front-end design and
  - documentation of the technical aspects covered and justification of the expected performance metrics for each of the functions deployed
- it is going to be crucial and adequate planning, time and resources estimation, organization and division of labor among all the different members of the team, as well as some basic sensible implementation of a correct CI/CD methodology based on **progressive** prototyping, testing and integration of each of the components seamlessly
- So, plan ahead, explore in advance potential bottlenecks in the process and distribute efforts and resources proportionately
  - General advice: start simple (in terms of initial features and model complexity as well), keep establishing performance baselines and then improve from there as far as your models (and your available time) allows

Presentation

- [Time: 30%] Explanation of the process followed
  - Tasks/Roles and organization of the team
  - The main findings and lessons learned
  - The technical approach for solving each of the requested functionalities
- [Time: 50%] A live demonstration of the prototype developed
  - Be ready to address some “user” live questions/inputs from the professor or your peers
- [Time: 20%] Ideas for improvement/expansion

Assignment Description DELIVERABLES

- Documentation
  - readme.md and PPTX/PDF of the project presentation
  - Main Document
    - A detailed document covering all the required aspects of the solution
  - Presentation Document
    - An executive summary of the solution analyzed, but not a simple elevator pitch.
    - It must include an overview of the Business/Technical components applied and the explanation of the main issues addressed (FOR INSTRUCTIONAL PURPOSES OF THE REST OF THE CLASS)
    - Suited to a 12 minute presentation, including an online/real-time demo of the application built
- Code and Executables
  - Data, code, generated models and streamlit application
    - requirements.txt

Note: all deliverables must be posted/sent by the Coordinator before the Presentation day

Administrative Notes

- Instructions for GROUP ASSIGNMENT 18.00, which is due one day prior to SESSION 29/30 (Dec 10th)
- Groups Assigned → based on your preferences informed to the application
- For SESSION 29/30:
  - Each **group** will nominate a Coordinator (responsible for sending the assignment's documents or materials)[only the Coordinator is required to send the document on behalf of the other members of the Group]
  - One presentation per team will be made → ~12 minutes each
  - Presentations must be made collaboratively by all members of the team
  - ~5 minutes for a joint discussion and questions about the projects presented for each group
  - 10~20 minutes for wrap-up and closing of the Course
- The work will be graded following the guidelines described later

SUBMIT YOUR WORK AS A STANDALONE FILE BY ZIPPING TOGETHER IN A SINGLE .ZIP FILE ALL THE PRESENTATIONS, CODE, MODELS, DATA AND ANY ADDITIONAL FILE REQUIRED FOR EXECUTING THE PROJECT AND REVIEWING THE DOCUMENTATION

NAME YOUR SUBMISSION AS "DAI_18_00 – Group {Group number xyz}.zip"

Notes for Grading

- The group presentations will be graded following the guidelines described
- **All members** of the group must participate collaboratively in the presentation in a freely organized/structured way
- Questions will be addressed by turn to every member of the group about questions related to the part or section he/she exposed or eventually about any other section or about general aspects regarding the concepts, theory, algorithms, technical or practical issues involved in the project presented (and related to the contents of the course) [an outline similar to that of a traditional oral exam]
- There will be an "*average*" grading for the Team as a whole, but optionally and only if circumstances require or advise it, **additional** points may be assigned to certain individuals within the team if disparities in contribution levels or knowledge regarding the project are detected. This will be probably an exceptional case but it is a prerogative of the evaluator.

Evaluation Guidelines

The overall evaluation of the project would be approximately:

- PROJECT CODE AND DOCUMENTATION 65%
- PRESENTATION AND FINAL PRODUCT QUALITY 35%

Use the following values just as an orientation for the amount of work (and extension) expected in your final workload and document.
At this stage we will be more concerned with the methodological, technical and functional aspects of the project (specially those related to the ML nature of the project) than with the general or visual appearance or usability of the final app developed

Evaluation Criteria and Rubric for ML Web App Development Project	%
Compliance with Functional Requirements	15
Acceptable and Justified Performance Metrics & Optimization	13
Smart and value-oriented data preparation and feature engineering	17
Adequate model training, validation and selection	10
Code Quality, Readability & Structure (Aimed for Maintainability and Improvement)	10
Sufficient Testing & Debugging	5
Innovation & Creativity	5
User Interface (UI) & User Experience (UX)	5
Presentation & Documentation	15
Team Collaboration & Project Management	5

Orientative only Rubrics

		0%	25%	50%	75%	100%	
Factors	**Weight**	**E**	**D**	**C**	**B**	**A**	**Value**
WEB APPLICATION ASSIGNMENT							
Evaluation Guidelines	**100**						**93.25**
Compliance with Functional Requirements	15.0						15.0
Acceptable and Justified Performance Metrics & Optimization	13.0						13.0
Smart and value-oriented data preparation and feature engineering	17.0						12.8
Adequate model training, validation and selection	10.0						10.0
Code Quality, Readability & Structure (Aimed for Maintainability and Improvement)	10.0						10.0
Sufficient Testing & Debugging	5.0						3.8
Innovation & Creativity	5.0						5.0
User Interface (UI) & User Experience (UX)	5.0						3.8
Presentation & Documentation	15.0						15.0
Team Collaboration & Project Management	5.0						5.0
	100.0						**93.3**
GRADING	***180***						***167.5***