Organizing your project directories in a professional manner is crucial for scalability, readability, and collaboration. Here’s a recommended directory structure for a data science project (like classification with an academic success dataset) following industry best practices:

**1. High-Level Project Structure**

bash

Copiar código

├── data/

│ ├── raw/ # Raw data files from Kaggle or other sources

│ ├── processed/ # Cleaned and processed datasets ready for analysis/modeling

├── notebooks/

│ ├── EDA.ipynb # Exploratory Data Analysis notebook

│ ├── Preprocessing.ipynb # Data preprocessing steps (cleaning, feature engineering)

│ ├── Modeling.ipynb # Model training, validation, and tuning

│ ├── Evaluation.ipynb # Model evaluation, metrics, and analysis

├── src/

│ ├── data/ # Scripts to load, preprocess, and save data

│ │ └── data\_loader.py # Functions to load raw data and output cleaned data

│ └── models/ # Scripts for model building, training, and evaluation

│ └── train\_model.py # Code for model training

│ └── evaluate\_model.py # Code for evaluating models

├── tests/

│ └── test\_data.py # Unit tests for data loading, preprocessing

│ └── test\_models.py # Unit tests for models

├── reports/

│ └── figures/ # Plots and figures generated during analysis

│ └── summary.pdf # Final report of analysis and results

├── environment.yml # Conda or pip environment dependencies

├── README.md # Project overview, instructions, and documentation

├── .gitignore # To ignore unnecessary files from version control

**2. Breakdown of Each Folder**

**1. data/**

* **Purpose**: To store all the data used in the project.
* **Sub-folders**:
  + raw/: Contains the original, unprocessed data. You shouldn't modify this data.
  + processed/: Holds the cleaned, transformed data ready for analysis or modeling.
* **Example files**: raw/student\_performance.csv, processed/student\_performance\_clean.csv.

**2. notebooks/**

* **Purpose**: Store Jupyter notebooks that track the experimentation process.
* **Typical Notebooks**:
  + **EDA.ipynb**: Contains exploratory data analysis such as visualizations, summary statistics, and initial findings.
  + **Preprocessing.ipynb**: Contains data cleaning, feature engineering, and preparation steps.
  + **Modeling.ipynb**: Model training, hyperparameter tuning, and validation experiments.
  + **Evaluation.ipynb**: Model evaluation metrics and comparison of different models.

**3. src/**

* **Purpose**: Holds Python scripts for loading data, processing it, and creating models.
* **Sub-folders**:
  + data/: Scripts related to data ingestion, cleaning, and transformation.
  + models/: Scripts related to model creation, training, tuning, and evaluation.
* **Key Files**:
  + data\_loader.py: Contains functions to load and preprocess the data.
  + train\_model.py: Scripts for training the classification models.
  + evaluate\_model.py: Code for evaluating models using metrics like accuracy, F1-score, etc.

**4. tests/**

* **Purpose**: Contains unit tests to ensure code correctness.
* **Key Files**:
  + test\_data.py: Contains tests for data loading and processing functions.
  + test\_models.py: Contains tests for model training and evaluation logic.

**5. reports/**

* **Purpose**: Stores final reports, figures, and visualizations generated from the analysis.
* **Sub-folders**:
  + figures/: Visualizations (e.g., confusion matrices, ROC curves) generated during the analysis.
  + summary.pdf: The final analysis report summarizing findings, methodology, and results.

**6. environment.yml**

* **Purpose**: Lists dependencies required for the project, used to create a reproducible environment with conda or pip.
* **Example**:

yaml

Copiar código

name: academic\_success\_classification

dependencies:

- python=3.8

- pandas

- scikit-learn

- matplotlib

- seaborn

**7. README.md**

* **Purpose**: Provides a description of the project, including its purpose, instructions for setup, and an overview of the contents. This is crucial for documentation and sharing with others.
* **Example**:

markdown

Copiar código

# Academic Success Classification

This project classifies students based on their academic performance using a dataset from Kaggle. The main objective is to predict student success based on various factors like study time, absences, family background, and more.

## Project Structure

- `data/`: Contains raw and processed data.

- `notebooks/`: Contains Jupyter notebooks for EDA, preprocessing, and modeling.

- `src/`: Scripts for data loading, model training, and evaluation.

- `reports/`: Visualizations and final reports.

## Getting Started

1. Clone the repository

2. Install dependencies using `conda env create -f environment.yml`

3. Run the notebooks in the order listed above.

**8. .gitignore**

* **Purpose**: This file ensures that unnecessary files (like data files, environment files, temporary files) are not tracked in version control (Git).
* **Example**:

plaintext

Copiar código

\_\_pycache\_\_/

.ipynb\_checkpoints/

data/raw/

data/processed/

.env

**3. Version Control (Git)**

* You can use git to track your progress and manage versions of the project.
* Regular commits with clear messages (e.g., git commit -m "Added EDA notebook with initial findings") will help you stay organized.

**4. Automation and Reproducibility**

* Use scripts for repetitive tasks (e.g., running preprocessing, training the model).
* Use tools like Makefile or DVC (Data Version Control) for versioning data and automating workflows.

**Conclusion**

This directory structure is designed for a flexible, modular, and scalable project that can be shared with others, reproduced, and expanded upon. It makes it easy for collaborators (or even your future self!) to understand the flow of your project.