AI Mastery Capstone - Project 1: AI Programming Foundations Project

Frank Allen Motley

Udacity Institute of AI & Technology

Master of Science in Artificial Intelligence
AI Mastery Capstone

Instructor

Due Date

AI Mastery Capstone - Project 1: AI Programming Foundations Project

Overview

This project builds a complete, reproducible data workflow in a Jupyter Notebook using Python, NumPy, Pandas, and Matplotlib/Seaborn. The notebook loads and analyzes the Titanic dataset from the seaborn example-data repository (Waskom, n.d.). The workflow demonstrates modular cleaning, exploratory analysis, visualization, and interpretation suitable as a foundation for future machine learning work.

Dataset Description

The Titanic dataset contains passenger-level records with survival outcome and demographic and travel attributes. In this workflow, the dataset has 891 rows and 15 columns, including variables such as survived, sex, class/pclass, age, and fare (Waskom, n.d.).

Missingness is concentrated in the deck field (688 missing) and age (177 missing), with small amounts of missingness in embarked and embark_town.

Workflow Description

Ingestion: The notebook loads the dataset from a stable CSV source and saves a local copy for repeatable runs.

Cleaning: Two reusable cleaning functions are applied. First, column names are standardized for consistency. Second, missing values are imputed using transparent rules (median for numeric columns and mode for categorical columns) based on common Pandas workflows for missing data handling (Pandas Development Team, n.d.).

Exploratory analysis: A reusable EDA function produces shape, dtypes, descriptive statistics, and grouped survival rates.

Visualizations: Three labeled plots are produced and saved to disk using Matplotlib/Seaborn conventions for titles and axis labels (Matplotlib Development Team, n.d.).

Summary: Findings and limitations are documented in the notebook and summarized here for the module report.

Key Decisions and Assumptions

Missing data handling is a high-impact decision. For this introductory workflow, median/mode imputation was selected because it is simple to explain and produces a fully analyzable table without dropping many rows. This choice assumes that missing values can be approximated without strongly distorting the distribution; in practice, alternative strategies (e.g., stratified imputation or modeling-based imputation) should be evaluated.

EDA focus prioritized interpretable group comparisons (survival by sex and passenger class) and distribution inspection (age). These were chosen because they align with the dataset's well-known structure and provide clear examples of aggregation and visualization.

Plots were designed to be publication-ready: each includes a title and labeled axes, and each is exported as a figure file for referencing in the written report (Matplotlib Development Team, n.d.).

Results and Interpretation

Figure 1 shows that survival rates differ substantially by sex, with female passengers exhibiting higher mean survival than male passengers. Figure 2 shows survival varies by passenger class, with first class highest and third class lowest, suggesting strong stratification in outcomes. Figure 3 compares age distributions by survival outcome; the distributions overlap considerably, implying that age alone may not explain survival and may interact with other variables such as class and sex.

These results are descriptive and do not establish causality. They are intended to demonstrate a sound workflow and clear communication of exploratory findings.

Responsible Practice (Bias and Data Quality)

Cleaning choices can introduce bias or misleading conclusions. For example, median imputation for age reduces variability and can change subgroup comparisons if missingness is not random. Similarly, dropping rows with missing values could disproportionately remove certain passenger groups, shifting estimated survival rates.

To reduce risk in this project, assumptions were kept explicit, and imputation rules were applied consistently. A next step would be to document the dataset's collection context and potential ethical considerations using structured dataset documentation practices (Gebru et al., 2021).

Reproducibility

Reproducibility is supported by a single notebook that runs top-to-bottom, a pinned environment snapshot in requirements.txt, and version control in Git. The intended Git workflow includes multiple commits reflecting each project stage (ingestion, cleaning, EDA, visualization, documentation) and at least one additional branch for development work, consistent with common Git best practices (Chacon & Straub, 2014).

A reviewer can reproduce the work by installing dependencies with pip install -r requirements.txt and executing data_workflow.ipynb in Jupyter.

References

Chacon, S., & Straub, B. (2014). Pro Git (2nd ed.). Apress. https://doi.org/10.1007/978-1-4842-0076-6

Pandas Development Team. (n.d.). Working with missing data. pandas documentation. Retrieved January 29, 2026, from https://pandas.pydata.org/docs/user_guide/missing_data.html

Matplotlib Development Team. (n.d.). Quick start guide. Matplotlib documentation. Retrieved January 29, 2026, from https://matplotlib.org/stable/users/explain/quick_start.html

Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., Daumé III, H., & Crawford, K. (2021). Datasheets for datasets. Communications of the ACM, 64(12), 86–92. https://doi.org/10.1145/3458723