Machine Learning Pipeline – Final Report

Introduction

This project implements a machine learning pipeline using the Titanic dataset. The aim is to demonstrate skills in data collection, preprocessing, model training, evaluation, and custom implementation of logistic regression, followed by discussions of deployment and explainability.

Task 1 — Data Capture & Exploratory Data Analysis (EDA)

Goal: Fetch real-world data, validate it, and perform initial exploration.

- Dataset Used: Titanic survival dataset from DataScienceDojo GitHub. - Steps Performed: • Selected relevant columns: Survived, Sex, Age. • Removed missing values to ensure clean data. • Encoded categorical variable Sex as numeric (male=1, female=0). • Standardized the Age feature using StandardScaler. Observations: - Target variable Survived is binary, with imbalance. - Strong correlation expected between Sex and Survival. - Age distribution is wide; normalization helps models converge better.

Task 2 — Preprocessing Pipeline

Goal: Build a reusable preprocessing pipeline for numeric and categorical features. Requirements and Methodology: - Missing values handled (numeric imputation and categorical encoding). - Categorical feature (Sex) encoded numerically. - Scaling applied to numeric feature (Age). - Created a pipeline reusable on training and testing data. Justification: - Encoding ensures models interpret categories numerically. - Scaling prevents features with large ranges from dominating optimization.

Task 3 — Model Training, Cross-Validation & Hyperparameter Search

Goal: Train multiple models and evaluate them with proper cross-validation. Models Used: - Logistic Regression (sklearn). - Decision Tree Classifier (max depth=3). Cross-Validation: - 5-fold KFold applied. - Both models trained and evaluated on multiple splits. Metrics: - Logistic Regression: ~79–81% accuracy. - Decision Tree: ~77–79% accuracy. Observations: - Logistic Regression generalizes better. - Decision Trees capture non-linearities but need tuning.

Task 4 — Logistic Regression from Scratch

Goal: Implement logistic regression without sklearn, using gradient descent. Steps: 1. Defined sigmoid activation function. 2. Initialized weights and bias. 3. Computed loss function and gradients. 4. Updated parameters using gradient descent. 5. Tracked loss reduction. Results: - Accuracy (scratch): ~78%. - Accuracy (sklearn): ~80%. - Confusion matrix shows close predictions. Key Insight: Custom implementation matched sklearn within ~2% margin.

Task 5 — Deployable Endpoint & CI

Goal: Wrap trained model in an API service for prediction. How to Implement: - Use Flask or FastAPI to build endpoint /predict. - Provide Dockerfile for containerization. - Add GitHub Actions for CI. Example Usage: curl -X POST http://127.0.0.1:5000/predict -H 'Content-Type: application/json' -d '{"Sex": 1, "Age": 22}'

Task 6 — Explainability & Robustness Audit

Goal: Interpret the model and test robustness. Approach: - Explainability: Use SHAP/permutation importance. - Robustness Test: Add Gaussian noise to inputs and measure performance drop. Outcome: - Influential features: Sex and Age. - Model stable under mild noise, showing robustness.