Project Overview

This project implements a machine learning pipeline to forecast employee performance. Three models are evaluated—**Linear Regression**, **Random Forest**, and **XGBoost**—and the best-performing one is deployed via a Flask API for real-time prediction use.GitHub

Key Components

1. Data Handling

- **Data Collection**: Import or ingest employee data containing relevant features.GitHub
- **Exploratory Data Analysis (EDA)**: Visualize feature distributions, analyze correlations, and derive descriptive statistics to uncover insights.GitHub

2. Data Preprocessing

- **Cleaning & Encoding**: Manage missing values and encode categorical variables into numeric form suitable for model consumption.GitHub
- **Feature Engineering**: Derive or transform features to improve model performance—e.g., combining, normalizing inputs.GitHub

3. Model Training & Evaluation

- **Models Used**:
  - **Linear Regression**
  - **Random Forest**
  - **XGBoost**
- **Evaluation Metrics**:
  - Mean Absolute Error (MAE)
  - Mean Squared Error (MSE)
  - R-squared (R^2) score

    These enable the comparison of model accuracy and performance.GitHub

4. Deployment

- **Flask API Integration**: The best-performing model is served via a Flask-based web API, allowing real-time predictions in production-like environments.GitHub

Summary Table

Component	Description
Models	Linear Regression, Random Forest, XGBoost
Preprocessing Steps	Data cleaning, encoding, feature engineering
Evaluation Metrics	MAE, MSE, R^2 score
Deployment	Best model wrapped in a Flask app for live predictions

Why It Matters

This project showcases a fully-functional end-to-end machine learning workflow—from raw data to deployment. It's ideal for demonstrating:

- **Model experimentation** (linear vs tree-based vs boosting).
- **Meaningful evaluation** through standard regression metrics.
- **Deployability**, by exposing the model via a web API for integration in larger systems.