**Technical Report: A Machine Learning System for Predicting Employee Productivity**

**Project Title:** Employee Performance Prediction

**Author:** Anupriya Gupta

**Date:** August 10, 2025

**Version:** 2.0

**1. Introduction**

**1.1. Problem Statement**

In manufacturing industries, particularly in sectors like garment production, operational efficiency is directly tied to workforce productivity. The ability to forecast an individual worker's output is critical for effective resource management, production planning, and identifying needs for targeted training. However, productivity is influenced by a complex interplay of variables, including work environment, scheduling, and financial incentives, making it difficult to predict through simple analysis alone.

**1.2. Project Goal**

The primary objective of this project is to design, develop, and deploy an end-to-end machine learning system capable of accurately predicting the productivity of garment workers. The system aims to transform raw operational data into actionable insights, delivered through an accessible web interface for use by managers and HR personnel.

**2. Methodology**

The project was executed following a structured data science lifecycle, encompassing data exploration, preprocessing, model development, and deployment.

**2.1. Data Sourcing and Exploration**

* **Dataset:** The project utilized the garments\_worker\_productivity.csv dataset, which contains 1197 records and 15 initial features.
* **Exploratory Data Analysis (EDA):** An initial EDA was performed to understand the data's structure and statistical properties. A correlation heatmap revealed initial relationships between numerical features like targeted\_productivity, smv, and the target variable, actual\_productivity.

**2.2. Data Preprocessing**

A series of data cleaning and transformation steps were essential to prepare the dataset for modeling:

1. **Handling Missing Data:** The wip (Work in Progress) column contained over 40% missing values and was deemed unreliable for imputation, leading to its removal from the dataset.
2. **Feature Engineering:** The date feature was decomposed to extract the month as a more general and useful feature. The original date column was then dropped.
3. **Data Standardization:** The department column contained duplicate categories due to whitespace inconsistencies ('finishing' vs. 'finishing '). These were consolidated into a single category.
4. **Categorical Data Encoding:** Non-numerical features (quarter, department, day) were transformed into a machine-readable integer format using a custom MultiColumnLabelEncoder.

**2.3. Model Development and Evaluation**

1. **Data Splitting:** The preprocessed dataset was partitioned into an 80% training set and a 20% testing set to ensure unbiased model evaluation.
2. **Model Selection:** Three regression algorithms were trained and evaluated:
   * **Linear Regression:** Served as a performance baseline.
   * **Random Forest Regressor:** An ensemble method chosen for its robustness.
   * **XGBoost Regressor:** A high-performance, gradient-boosted tree algorithm.
3. **Performance Metrics:** Models were compared based on their R2 Score, Mean Absolute Error (MAE), and Mean Squared Error (MSE).
4. **Model Choice:** The **XGBoost Regressor** demonstrated the highest R2 Score and the lowest error rates, establishing it as the superior model for this prediction task. The trained XGBoost model was then serialized into a gwp.pkl file for deployment.

**3. System Architecture**

The predictive model was deployed as a web application using the Flask framework, creating a clear separation between the backend logic and the frontend user interface.

**3.1. Backend Logic (app.py)**

* **Model Loading:** Upon application startup, the gwp.pkl file is loaded into memory.
* **Routing:** Flask routes are defined to handle HTTP requests for different pages (/, /about, /predict).
* **Prediction Endpoint:** The /predict route is configured to accept POST requests from the user input form. It retrieves the 13 input features, converts them into the required NumPy array format, and passes them to the loaded model.
* **Output Processing:** The model's numerical prediction is classified into one of three categories ("Averagely productive," "Medium productive," "Highly productive") before being sent to the results page.

**3.2. Frontend Interface (templates/)**

* **Templating:** A base.html template was created to maintain a consistent header, footer, and styling across all pages.
* **User Interface:**
  + home.html: The main landing page.
  + about.html: Provides context and details about the project.
  + predict.html: A user-friendly form for data entry, with labels indicating expected input ranges.
  + result.html: Displays the final, classified prediction to the user.
* **Styling:** The interface was styled using **Tailwind CSS** for a modern, clean, and responsive design.

**4. User Guide**

To set up and run the project locally, follow these steps:

1. **Clone the repository** and navigate into the project folder.
2. **Create and activate a Python virtual environment.**
3. python -m venv venv
4. source venv/bin/activate # On macOS/Linux
5. # .\venv\Scripts\activate # On Windows
6. **Install the required dependencies.**
7. pip install flask numpy pandas scikit-learn xgboost
8. **Run the application.**
9. # Navigate to the Flask sub-directory
10. cd Flask
11. python app.py
12. **Access the application** by opening a web browser and navigating to http://127.0.0.1:5000.

**5. Conclusion and Future Work**

This project successfully developed an end-to-end machine learning solution that transforms raw data into actionable business intelligence. The deployed web application provides an intuitive interface for non-technical users to leverage a powerful predictive model.

**Potential future enhancements could include:**

* **Model Retraining Pipeline:** Implementing a mechanism to automatically retrain the model with new data to prevent model drift.
* **Advanced Feature Engineering:** Incorporating additional data points, such as worker tenure or historical absenteeism, could further improve accuracy.
* **Dashboarding:** Adding a dashboard to visualize historical predictions and track team performance over time.