Project Report - Phase 1: Initialization and Planning

Author: Arpit Jain **Date:** August 19, 2025 **Project:** Employee Productivity Prediction

1.1 Introduction and Project Vision

This document initiates the **Employee Productivity Prediction** project. The vision is to leverage machine learning to forecast the productivity of garment industry workers with high accuracy. In a domain where efficiency drives competitiveness, such predictions can inform strategic planning, resource allocation, and performance optimization. This phase defines the project's scope, objectives, and technical roadmap.

1.2 Problem Statement and Business Case

Problem. Garment manufacturing runs on tight margins and schedules. Without reliable productivity forecasts, organizations face missed deadlines, suboptimal staffing, and reduced profitability. This project addresses the need for a data-driven approach to predict **actual productivity**.

Business Case. A robust predictive model will enable:

- **Improved Production Planning:** More accurate forecasts for scheduling and target setting.
- **Enhanced Resource Management:** Insights into drivers of productivity to optimize team composition and workflow.
- **Proactive Performance Management:** Early identification of teams or individuals needing support, training, or incentives.

1.3 Project Objectives and Scope

Objectives

- Conduct an in-depth analysis of the **garments_worker_productivity.csv** dataset.
- Develop and compare multiple ML models to select the most accurate predictor.
- Build a user-friendly web application for interactive predictions.
- Deploy the application on a public cloud platform for real-world use.

Scope

Covers the full ML lifecycle: data exploration, preprocessing/feature engineering, model training and evaluation, and deployment as a web service.

1.4 Technology Stack Selection

- **Language:** Python 3
- **Core Libraries:** Pandas, NumPy, scikit-learn, XGBoost
- **Web Framework:** Flask
- **Deployment:** Render with Gunicorn as the WSGI server

This stack is mature, well-supported, and suitable for both machine learning and web delivery.

Project Report - Phase 2: Data Collection and Preprocessing

Author: Arpit Jain **Date**: August 20, 2025 **Project:** Employee Productivity Prediction

2.1 Data Sourcing and Initial Exploration

The project is built on the garments_worker_productivity.csv dataset, which contains 1,197 records and 15 attributes describing various aspects of employee performance in a garment factory. An exploratory data analysis (EDA) was conducted to:

- Understand the dataset's structure, data types, and statistical properties.
- Examine the distribution of key variables.
- Identify potential data quality issues requiring remediation.

This initial exploration guided the subsequent data preprocessing strategy.

2.2 Data Cleaning and Transformation

To ensure the dataset was suitable for machine learning, the following preprocessing steps were applied:

- Standardization of Categorical Data: The department column included inconsistencies such as "sweing" and "finishing ". These were corrected to "sewing" and "finishing" to maintain categorical integrity.
- Handling of Missing Values: The wip (work in progress) column contained a substantial number of missing entries. To preserve dataset quality, this feature was dropped from the final model input.
- Feature Engineering: The date column was transformed to derive a new month feature, capturing potential seasonal effects on productivity. The original date column was then removed.

2.3 Categorical Data Encoding

Since machine learning models require numerical inputs, categorical attributes (quarter, department, and day) were converted into numerical form. Label Encoding was applied, mapping each unique category within a feature to an integer value. This ensured the dataset was fully compatible with the modeling algorithms.

2.4Final Dataset Preparation

After cleaning and encoding, the dataset was finalized for model development. Key steps included:

- Splitting the dataset into features (X) and the target variable (y = actual_productivity).
- Ensuring a clean, structured dataset to support effective training, validation, and evaluation of multiple machine learning models in the subsequent phase.

Project Report - Phase 3: Model Development

Author: Arpit Jain **Date:** August 21, 2025 **Project:** Employee Productivity Prediction

3.1. Model Selection and Rationale

For this regression task, three different machine learning models were selected for training and comparison:

- **Linear Regression:** Chosen as a simple baseline model to establish a benchmark for performance.
- **Random Forest Regressor:** An ensemble learning method that is known for its high accuracy and ability to handle complex relationships in data.
- **XGBoost Regressor:** A highly optimized and powerful gradient boosting algorithm, which is often a top performer in machine learning competitions.

This selection provides a good range of models, from a simple baseline to more complex and powerful ensemble methods.

3.2. Training and Testing Methodology

The preprocessed dataset was split into a training set (80% of the data) and a testing set (20% of the data). This separation is crucial to ensure that the models are evaluated on data they have not seen before, providing an unbiased assessment of their performance. Each of the three models was trained on the same training data.

3.3. Model Performance Evaluation

The performance of each trained model was evaluated on the test set using the following standard regression metrics:

- **Mean Absolute Error (MAE):** This metric provides a straightforward measure of the average error of the model's predictions.
- **Mean Squared Error (MSE):** This metric penalizes larger errors more heavily.
- **R-squared (R^2) Score:** This is a key metric that indicates the proportion of the variance in the target variable that is predictable from the features. A higher R^2 score indicates a better fit.

3.4. Model Selection and Finalization

After comparing the evaluation metrics for all three models, the **XGBoost Regressor** was identified as the best-performing model, primarily due to its superior R^2 score. This indicates that it was the most accurate and reliable model for this particular dataset. The trained XGBoost model was then saved as a pickle file (best_model.pkl) so that it could be easily loaded and used in the web application without the need for retraining.

Project Report - Phase 4: Application Development and Deployment

Author: Arpit Jain **Date:** August 22, 2025 **Project:** Employee Productivity Prediction

4.1. Web Application Development

A web application was developed using the **Flask** framework in Python to provide a userfriendly interface for the machine learning model. The application consists of a singlepage interface with a form where users can input the various attributes of an employee.

- **Frontend:** The user interface was built with HTML and styled using **Tailwind CSS** to create a modern and responsive design.
- **Backend:** The Flask backend handles the form submission, preprocesses the user's input to match the format expected by the model, and then uses the loaded .pkl model to make a prediction.

4.2. Application Logic and Flow

1. The user navigates to the application's URL.
2. They fill out the web form with the required employee data.
3. Upon clicking "Predict Productivity," the data is sent to the Flask server.
4. The server's Python code applies the same **Label Encoding** transformations to the categorical data as was done during model training.
5. The processed data is then passed to the XGBoost model for prediction.
6. The model returns a productivity score, which is then categorized as "High," "Medium," or "Low" and displayed back to the user on the web page.

4.3. Deployment to Render

The final application was deployed to the **Render** cloud platform to make it publicly accessible. The deployment process involved:

1. Pushing all project files (app.py, best_model.pkl, requirements.txt) to a GitHub repository.
2. Creating a new Web Service on Render and linking it to the GitHub repository.
3. Configuring the build command (pip install -r requirements.txt) and the start command (gunicorn app:app).

The use of a requirements.txt file ensures that all necessary Python libraries are installed in the deployment environment, and Gunicorn is used as a production-ready web server.

4.4. Conclusion and Future Work

This project successfully demonstrates the end-to-end process of building and deploying a machine learning application. The final deployed web service provides a valuable and accessible tool for predicting employee productivity.

Future enhancements could include:

- **Hyperparameter Tuning:** To further improve the model's accuracy.
- **Data Visualization:** Adding charts and graphs to the web interface to provide more insights.
- **Batch Predictions:** Allowing users to upload a CSV file to get predictions for multiple employees at once.