Employee Productivity Prediction - Final Project Report

# 1. Introduction

## 1.1 Project Overview

This project focuses on predicting employee productivity using machine learning techniques. The web application developed using Flask allows users to upload historical productivity datasets, train machine learning models, visualize performance comparisons, and predict productivity for new input data.

## 1.2 Objectives

- Analyze employee productivity data  
- Build predictive models (Linear Regression, Random Forest, XGBoost)  
- Provide a user-friendly interface for training models and predictions  
- Enable data visualization and model comparison

# 2. Project Initialization and Planning Phase

## 2.1 Define Problem Statement

To predict employee productivity based on factors like targeted productivity, SMV, WIP, overtime, incentives, idle time, and team-related data.

## 2.2 Project Proposal (Proposed Solution)

Develop a machine learning web application that preprocesses data, trains models, evaluates them, and provides predictions in a visual and interactive format.

## 2.3 Initial Project Planning

We planned to use Flask for the web interface and scikit-learn/xgboost for model building. The dataset is taken from the Kaggle Garments Worker Productivity dataset.

# 3. Data Collection and Preprocessing Phase

## 3.1 Data Collection Plan and Raw Data Sources Identified

Dataset source: Kaggle - Garments Worker Productivity dataset.

## 3.2 Data Quality Report

Data cleaning included removing nulls, encoding categorical columns, and handling missing values using SimpleImputer.

## 3.3 Data Exploration and Preprocessing

Preprocessing steps:  
- Drop unused columns like date  
- Label encode categorical features (quarter, department, day)  
- Scale numeric features  
- Split into training and testing sets

# 4. Model Development Phase

## 4.1 Feature Selection Report

All relevant features after preprocessing were retained, as no features were redundant after encoding.

## 4.2 Model Selection Report

The models considered were Linear Regression, Random Forest, and XGBoost.

## 4.3 Initial Model Training Code, Model Validation and Evaluation Report

The models were trained using an 80-20 train-test split. Metrics such as MAE, RMSE, and R² were used to evaluate the models.

# 5. Model Optimization and Tuning Phase

## 5.1 Hyperparameter Tuning Documentation

Random Forest and XGBoost models can be further optimized with GridSearchCV/RandomizedSearchCV, but initial results were satisfactory for deployment.

## 5.2 Performance Metrics Comparison Report

Random Forest and XGBoost performed better than Linear Regression.

## 5.3 Final Model Selection Justification

The final model recommended for deployment was XGBoost due to its superior accuracy.

# 6. Results

## 6.1 Output Screenshots

The web application displays:  
- A table of metrics  
- A bar chart comparing MAE, RMSE, R²  
- A heatmap for metric correlations  
- A prediction form for new data

# 7. Advantages & Disadvantages

Advantages:  
- Automates model training  
- User-friendly web interface  
- Visualization support  
  
Disadvantages:  
- Requires structured CSV input  
- Models are retrained every upload

# 8. Conclusion

The project demonstrates how machine learning can assist organizations in predicting employee productivity and planning workforce allocation.

# 9. Future Scope

- Add cloud deployment  
- Implement user authentication  
- Use more advanced ML models and hyperparameter tuning  
- Add dashboards for analysis

# 10. Appendix

## 10.1 Source Code

The source code is available in the project repository.

## 10.2 GitHub & Project Demo Link

GitHub Repository: https://github.com/sunidhi1dec/employee-productivity-flask