Machine Learning Approach for Employee Performance Prediction

Table of Contents

1. Introduction

1.1 Project Overview

This project presents a machine learning-based system to predict employee performance in a garment manufacturing environment. Leveraging structured historical data, the goal is to build a predictive model that analyzes employee-related metrics and forecasts their productivity. The system aims to help organizations optimize resource allocation, enhance productivity, and support talent retention and performance management efforts.

1.2 Objectives

- Predict actual employee productivity using historical performance data
- Assist HR and management in identifying high and low-performing individuals
- Aid in data-driven resource allocation and training recommendations
- Integrate the model into a web-based application for easy usability

2. Project Initialization and Planning Phase

2.1 Define Problem Statement

Employee productivity in the manufacturing sector is difficult to monitor and predict using manual methods. This project aims to develop a machine learning solution that predicts productivity based on past performance indicators and organizational data. The solution focuses on addressing inefficiencies in resource utilization and enabling proactive performance management.

Employee Performance Predictor Problem Statement Report:*click here*

2.2 Project Proposal (Proposed Solution)

The proposed system integrates historical employee data, including department, team, SMV (Standard Minute Value), incentives, and other work-related inputs to train a machine learning model. The model is deployed through a web interface, allowing users to input employee features and obtain a performance prediction. The platform provides predictive insights for HR and operational decision-makers.

Employee Performance Predictor Project Proposal Report:*click here*

2.3 Initial Project Planning

The project follows a 5-step pipeline:

1. Data Collection
2. Data Preprocessing and Visualization
3. Model Building
4. Model Evaluation and Saving
5. Deployment via a Flask Web Application

A structured timeline was followed for implementation, model integration, and report generation.

Employee Performance Predictor Project Planning Report:*click here*

3. Data Collection and Preprocessing Phase

3.1 Data Collection Plan and Raw Data Sources Identified

The dataset was sourced from Kaggle:
"Garments Worker Productivity"
The dataset includes 1,197 records across 15 features, such as:

- Department
- Day
- Team
- Targeted and Actual Productivity
- Overtime
- Incentives
- SMV (Standard Minute Value)

Employee Performance Predictor Data Collection Report:*click here*

3.2 Data Quality Report

- Null values found in the `wip` column (506 missing entries)
- The dataset was otherwise clean with well-structured columns
- Minor categorical inconsistencies (e.g., "finishing" appearing twice) were resolved by string stripping

Employee Performance Predictor Data Quality Report:*click here*

3.3 Data Exploration and Preprocessing

- Performed correlation analysis using a heatmap
- Descriptive statistics and type analysis via `.describe()` and `.info()`
- Removed unnecessary columns (`date`, `idle_men`, `no_of_style_change`)
- Applied one-hot encoding on categorical features (`department`, `quarter`, `day`)
- Split the data into feature matrix (X) and target vector (y)
- Saved feature order for model serving via Flask

Employee Performance Predictor Data Exploration and Pre-Processing Report:*click here*

4. Model Development Phase

4.1 Feature Selection Report

Based on correlation and practical relevance, the following features were retained:

- `smv`, `wip`, `incentive`, `over_time`, `no_of_workers`, and encoded categorical columns

Employee Performance Predictor Feature Selection Report:*click here*

4.2 Model Selection Report

Models tested:

- Linear Regression
- Random Forest Regressor (Selected)
- Gradient Boosting Regressor
  Random Forest performed best with low MAE and stable predictions, even with minor data inconsistencies.

Employee Performance Predictor Model Selection Report:*click here*

4.3 Initial Model Training Code, Model Validation and Evaluation Report

- Train-test split: 80–20
- Model trained on cleaned and encoded data
- Performance metrics:
  - Mean Absolute Error (MAE): ~0.08
  - R^2 Score: Satisfactory
- Model saved as `model.pkl`
- Feature order saved as `feature_order.pkl`

Employee Performance Predictor Model Development Phase Report:*click here*

5. Model Optimization and Tuning Phase

5.1 Hyperparameter Tuning Documentation

Grid search and random search were tested, but Random Forest’s default parameters already performed well. Minor improvements with n_estimators and max_depth tuning were explored but not deployed to maintain simplicity.

5.2 Performance Metrics Comparison Report

Model	MAE	R^2 Score
Linear Regression	~0.12	Low
Gradient Boosting	~0.09	Medium
Random Forest	**~0.08**	**Best**

5.3 Final Model Selection Justification

Random Forest was selected due to:

- Lower MAE
- Robust handling of categorical + numerical data
- Minimal tuning required
- Suitable for deployment in production-level apps

Employee Performance Predictor Model Optimization and Tuning Phase Report:*click here*

6. Results

6.1 Output Screenshots

Output Screenshots of website & video representation: *click here*

7. Advantages & Disadvantages

Advantages

- Accurate and scalable model for performance prediction
- Easy-to-use web interface
- Enables HR teams to make informed decisions

Disadvantages

- Data is specific to garment factory domain (limited generalization)
- No real-time integration with live HR databases
- Missing values in raw data required cleaning

8. Conclusion

The project successfully demonstrates how machine learning can be applied to predict employee productivity in a manufacturing setting. It shows potential for broader adoption in HR analytics systems. The final application offers a simple, effective way to test various employee profiles and receive performance predictions instantly.

9. Future Scope

- Deploy the application on cloud platforms (Heroku, Render, AWS)
- Integrate with live HR databases (e.g., PostgreSQL or HRMS tools)
- Add features like alert systems for low predicted productivity
- Improve model performance via feature engineering or neural networks

10. Appendix

10.1 Source Code

All source files: *click here*

10.2 GitHub & Project Demo Link

- **GitHub Link**: click here
- **Project Demo Video**: *click here*