PROJECT REPORT

Machine Learning Approach for Employee Productivity Prediction

Milestone 1: Data Collection

Activity 1: Data Collection Plan and Dataset Sourcing
Conducted comprehensive research to identify and gather a relevant dataset named `garments_worker_productivity.csv`. This dataset includes detailed productivity records from a garment manufacturing environment, capturing variables essential for modeling employee performance.

DataSet Link : Click_Here

Activity 2: Dataset Structure Review
Performed a thorough review of dataset columns such as department, day, quarter, number of workers, overtime, incentives, idle time, and work-in-progress (wip). Documented their meanings, data types, and roles in predicting actual productivity, ensuring the dataset met quality and completeness standards before moving to analysis.

Read:click

Milestone 2: Visualizing and Analyzing the Data

Activity 1: Importing Libraries
Imported critical Python libraries including pandas for data handling, seaborn and matplotlib for visualization, enabling detailed exploratory data analysis.

Requirements:Click_Here

Activity 2: Reading the Dataset
Loaded the dataset, examined its shape, column data types, and generated a snapshot of the first few records to confirm data integrity and understand overall structure.

Activity 3: Correlation Analysis
Conducted correlation analysis to identify relationships between input features and the target variable. This helped to highlight which factors like overtime or incentives could strongly influence employee productivity.

Activity 4: Descriptive Analysis and Visualization
Used summary statistics, heatmaps, and boxplots to detect trends, spot anomalies, and gain insights into feature distributions. This step provided a data-driven foundation for feature selection and preprocessing decisions.

Milestone 3: Data Pre-processing

Activity 1: Checking for Null Values
Scanned the dataset for missing values and addressed them by filling in neutral or calculated replacements to maintain data consistency.

Activity 2: Handling Date & Department Columns
Transformed categorical fields like department, day, and quarter into numerical form using Label Encoding to make them suitable for machine learning models.

Activity 3: Handling Categorical Data
Verified that all categorical data was properly encoded, ensuring uniform representation and avoiding errors during model training.

Activity 4: Splitting Data
Divided the cleaned dataset into training and testing subsets to evaluate model generalization capability. This practice ensures performance is measured on unseen data rather than the data the model was trained on.

Notebook:click_here

Milestone 4: Model Building

Activity 1: Importing Model Libraries
Integrated scikit-learn libraries necessary for training and evaluating regression models.

Activity 2: Initializing and Training the Model
Built and trained multiple regression models, including Linear Regression for baseline comparison, Decision Tree for interpretability, and Random Forest Regressor for advanced accuracy.

Activity 3: Evaluating Model Performance
Calculated R^2 scores and compared predictions to actual productivity values, ensuring models performed reliably on unseen data.

Activity 4: Saving the Model
Preserved the best-performing Random Forest model by saving it as `model.joblib` for deployment in the application phase.

model:click_here

Model	R^2 Score

Linear Regression	**~0.35**
Decision Tree	**~0.55**
Random Forest	**~0.65–0.70**

Final Selection: Choose Random Forest Regressor due to its superior accuracy and robustness compared to other models.

Milestone 5: Application Building

Activity 1: Creating an HTML Interface
Designed a simple yet intuitive HTML page to collect inputs such as department, day, number of workers, overtime hours, and incentives directly from users.

Activity 2: Developing Backend Python Code
Implemented Flask-based backend logic to receive form data, process it, load the trained model, and return predicted productivity to the frontend.

Activity 3: Integrating and Testing the App
Connected the HTML interface with the backend Python code, performed extensive local testing, and refined the user experience to ensure accurate, real-time predictions.
Source

Application:app

Results & Deliverables

- **Developed and validated a machine learning model with an R^2 score of approximately ~0.65–0.70.**
- **Built a Flask web application for end-user interaction and live productivity forecasting.**
- **Created comprehensive documentation detailing each milestone, design decision, and testing outcome.**

Source-Code:Github
OUTPUT: click_here

Future Scope

- **Enhance the system by integrating real-time data streams for dynamic prediction updates.**
- **Add interpretability tools such as SHAP to explain model predictions to users.**
- **Build advanced dashboards for visual insights.**
- **Experiment with additional machine learning models, including XGBoost or deep neural networks, to further improve accuracy.**

Conclusion

This project successfully delivered an end-to-end machine learning solution for predicting employee productivity in garment manufacturing. Following a milestone-driven approach ensured structured progress, resulting in a robust predictive model and user-friendly application. The project highlights the real-world value of machine learning in operational decision support.

Application Page:Main Page
Result Page:Predict Page
