Employee Performance Prediction UsingMachine Learning

Internship Project under Smart Intenz

Submitted by : Rohit Mishra

Milestone 1: Project Initialization and Planning Phase

Activity 1: Define Problem Statement

Problem Statement: In the competitive garment industry, efficient workforce utilization is vital. Managers often struggle to predictemployee productivity due to fluctuating workloads and task diversity. The challenge is to build a systemthat can analyzeinput metrics such as workinghours, team size, and task type to accurately predict productivity, enabling proactive resourceplanning.

Activity 2: Project Proposal(Proposed Solution)

The proposed projectaims to harness the power of machinelearning to build an accurateemployee productivity prediction system. Using a cleaned and preprocessed dataset from Kaggle, the model will be trained using advanced regression algorithms like XGBoost. The final output will be delivered via a web-based interface, allowing HR teams or managers to input daily metrics and receive productivity predictions in real time. This systempromotes data-driven decision-making, workforce optimization, and performance forecasting.

Activity 3: Initial Project Planning

Project planning began with requirement analysis and feasibility checks.Tasks were broken down into modules: data acquisition, cleaning, model training, evaluation, web integration, and testing. Tools selected included Python, Flask, Jupyter Notebook, Git, and GitHub. Timeline was divided into sprints focusing on data, model, and deployment.

Milestone 2: Data Collection and Preprocessing Phase

Activity 1: Data Collection and Sources

Data was collected from Kaggle's 'Productivity Prediction of Garment Employees' dataset. The dataset included over 1100 rows and featured variables such as 'smv', 'over_time', 'team', 'no_of_workers', and more. Data integrity was validated by checking for nulls, duplicates, and out-of- range values.

Activity 2: Data Quality Assessment

Missing values were imputed, and categorical data was encoded using label encoding. Outliers were handled using IQR and z-score methods. The dataset was then normalized to ensure consistency during model training. A data profiling report was also generated using Pandas Profiling to verify the balance and distribution.

Activity 3: Exploratory Data Analysis and Preprocessing

EDA revealed strongcorrelations between 'smv','over_time', and actualproductivity. Feature engineering included generating a new feature called 'worker_efficiency'. Categorical columns were encoded using LabelEncoder, and the entiredataset was split into train and test sets usingan 80-20 split.

Milestone 3: Model Development Phase

Activity 1: Feature Selection and Engineering

Features were selected based on correlation heatmaps and importance ranking from the XGBoost model.Low-variance and redundant features were removed. Engineered features like worker_efficiency significantly improved model performance.

Activity 2: Model Selectionand Evaluation

Multiple algorithms were tested: Linear Regression, Random Forest, Decision Tree, and XGBoost. XGBoost outperformed otherswith an R^2 score of 0.90 and MAE under 0.05. Evaluation metrics included Mean Squared Error, Mean Absolute Error, and R^2 Score.

Activity3: Model Training and Exporting

After finalizing hyperparameters using GridSearchCV, the XGBoost model was trained and validated. The trained model was serialized using 'joblib' for deployment in the Flask web application.

Milestone 4: Model Optimization and Tuning Phase

Activity 1: Hyperparameter Tuning

GridSearchCVwas used to tune key XGBoosthyperparameters: max_depth, learning_rate, and n_estimators. Cross-validation ensured robust tuning. Final model yieldedreduced RMSE and improved R^2.

Activity 2: Performance Comparison

Baseline vs TunedModel:

1. R^2 improved from 0.86 to 0.90
2. MAE reduced from 0.07 to 0.04

This confirmed model reliability and readiness for deployment.

Activity 3: Final Model Justification

TheXGBoost model was chosen for its superiorperformance, speed, and handling of complex feature interactions. It generalizes well and consistently produces accurate predictions.

Milestone 5: Web Deployment and Documentation

The model was deployed using Flask, integrated with a responsivefrontend designed using HTML and CSS. Inputs were captured through a form, passed to the backend,and the prediction was rendered instantly. The app was testedlocally and version-controlled using GitHub.

Milestone 6: Project Demonstration

A demonstration video was recorded explaining:

3. Dataset Overview
4. ML Pipeline
5. Web App Usage

This showcased the real-world utility and user-friendly interface of the application.