Garment Manufacturing: Predicting Employee Productivity

Erning Xu

Brown University

GitHub[1]

1. Introduction

Predicting worker productivity is a vital challenge in labor-intensive industries like garment manufacturing. Accurate productivity forecasts enable better resource allocation, improved operational efficiency, and informed decision-making. Traditional approaches often rely on heuristics or static metrics, which fail to capture the intricate relationships between workforce characteristics, task complexity, and operational conditions. These limitations highlight the need for machine learning models capable of uncovering hidden patterns in data and providing actionable, data-driven predictions.

The dataset used in this analysis comes from a garment manufacturing facility and includes a range of features relevant to worker productivity. Key variables include workforce attributes such as the number of workers, incentives, and department . Task complexity is captured through measures like the standard minute value and work in progress. Temporal information, including day and quarter, provides a context for time-dependent productivity trends. The target variable, actual_productivity, is a continuous measure of worker output. The dataset poses challenges such as missing values, correlated features, and non-linear relationships, making it an ideal candidate for machine learning solutions.

2. Exploratory Data Analysis

The analysis begins by examining the trend of actual productivity over time. The daily fluctuations in actual productivity are visualized alongside a linear regression trend line. Productivity ranges from 0.2 to 1.0, with significant variability observed on a daily basis. The trend line suggests a slight decline in productivity over the observation period, indicating potential operational inefficiencies or external factors affecting worker performance (Figure 1).

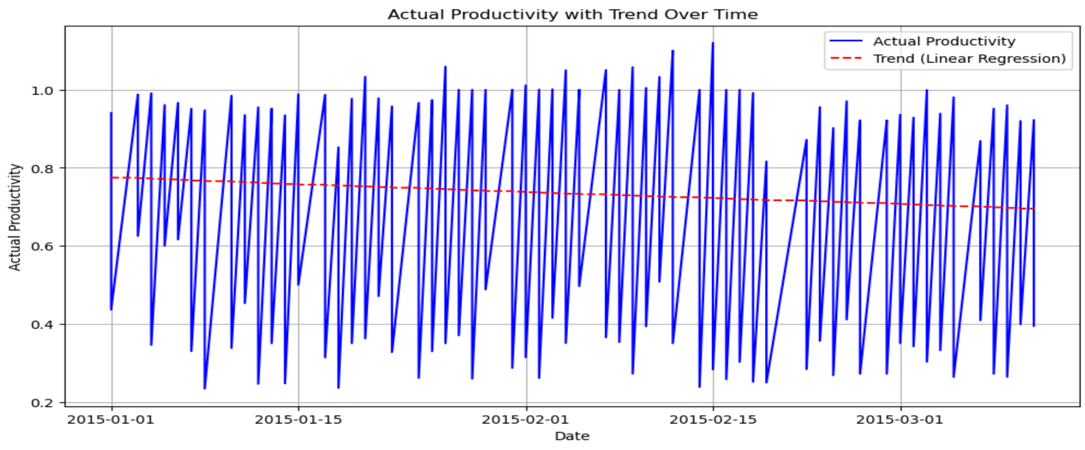

Figure 1. Productivity fluctuates significantly on a daily basis, with values ranging from 0.2 to 1.0.

Next, average productivity is analyzed by department and team. The finishing department consistently outperforms the sewing department in terms of productivity. Team-specific differences are also noticeable, with certain teams in both departments achieving higher productivity levels than others. This variation likely reflects differences in team dynamics, task allocation, and operational efficiency (Figure 2). When productivity is aggregated at the department level, the finishing department exhibits slightly higher average productivity (~0.75) compared to the sewing department (~0.72), suggesting that workflows in finishing may be more optimized (Figure 3).

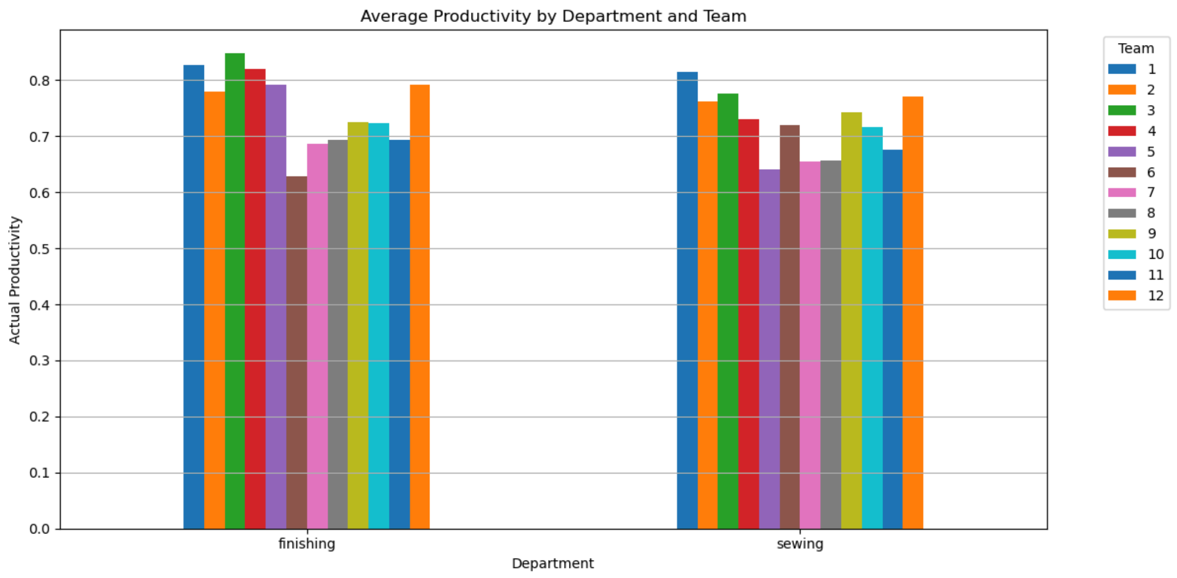

Figure 2. Team-specific differences are noticeable, with some teams consistently outperforming others.

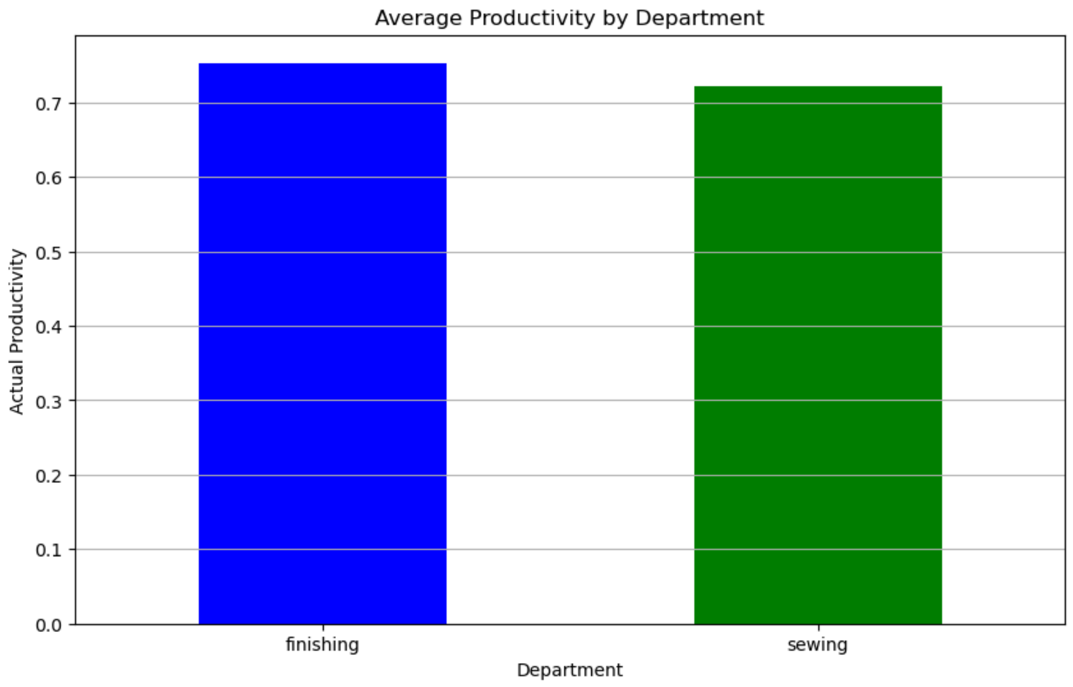

Figure 3. The finishing department generally has higher average productivity than the sewing department.

The correlation matrix provides insights into the relationships between features in the dataset. Targeted Productivity shows the strongest positive correlation with Actual Productivity (0.42), which might indicate that employees performed better when they are under pressure. Task complexity (smv) and overtime (over_time) are also highly correlated (0.67), indicating that more complex tasks often require additional time. Most other features show weak correlations with Actual Productivity, suggesting that non-linear modeling techniques may be needed to capture the underlying patterns (Figure 4).

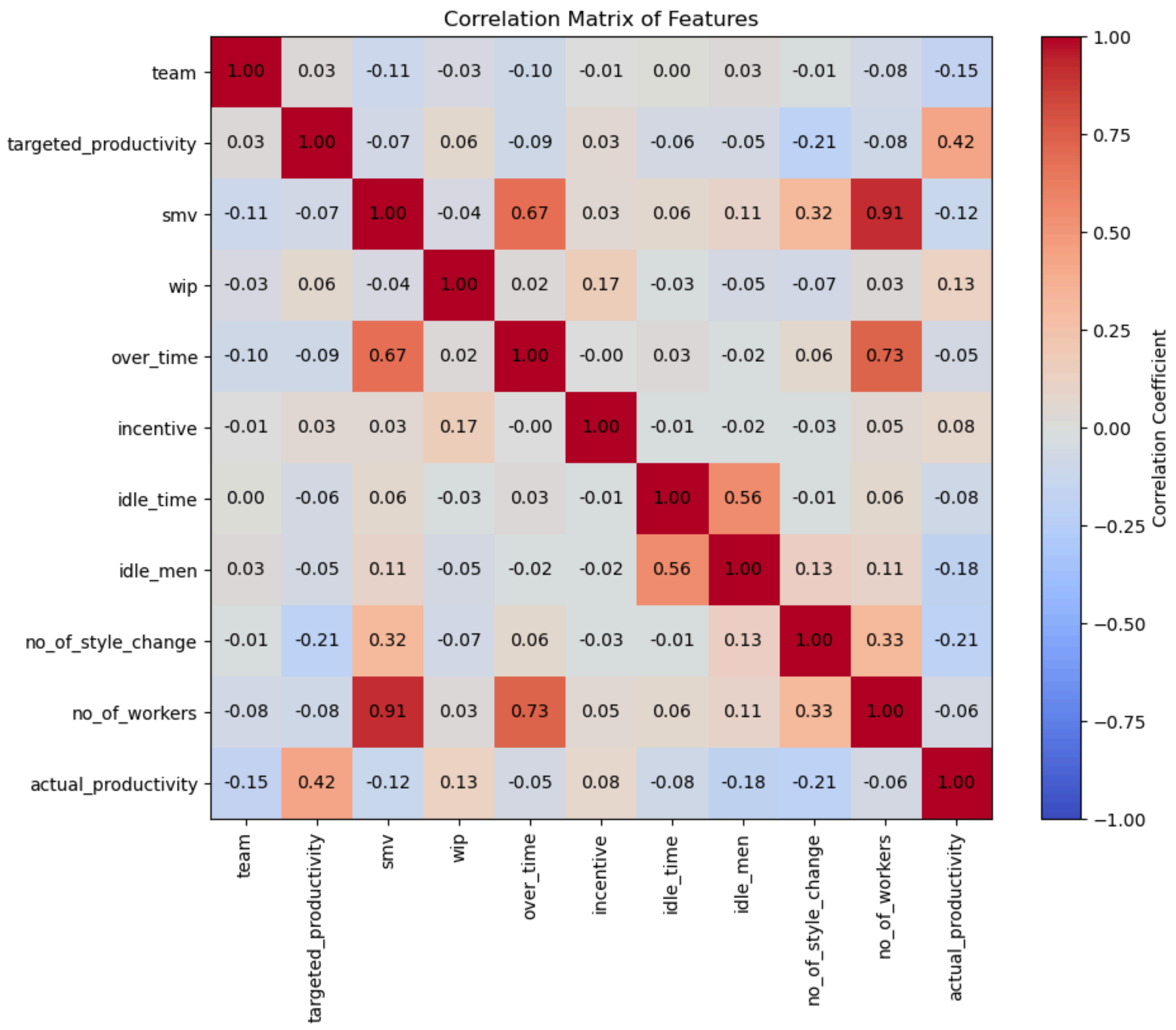

Figure 4. targeted_productivity has the strongest positive correlation with actual_productivity.

The distribution of Actual Productivity reveals that most values fall between 0.65 and 0.85, with a clear peak around 0.8. This distribution indicates that workers generally perform close to their productivity targets, with a few instances of very low productivity (<0.4) likely representing outliers or underperforming tasks (Figure 5).

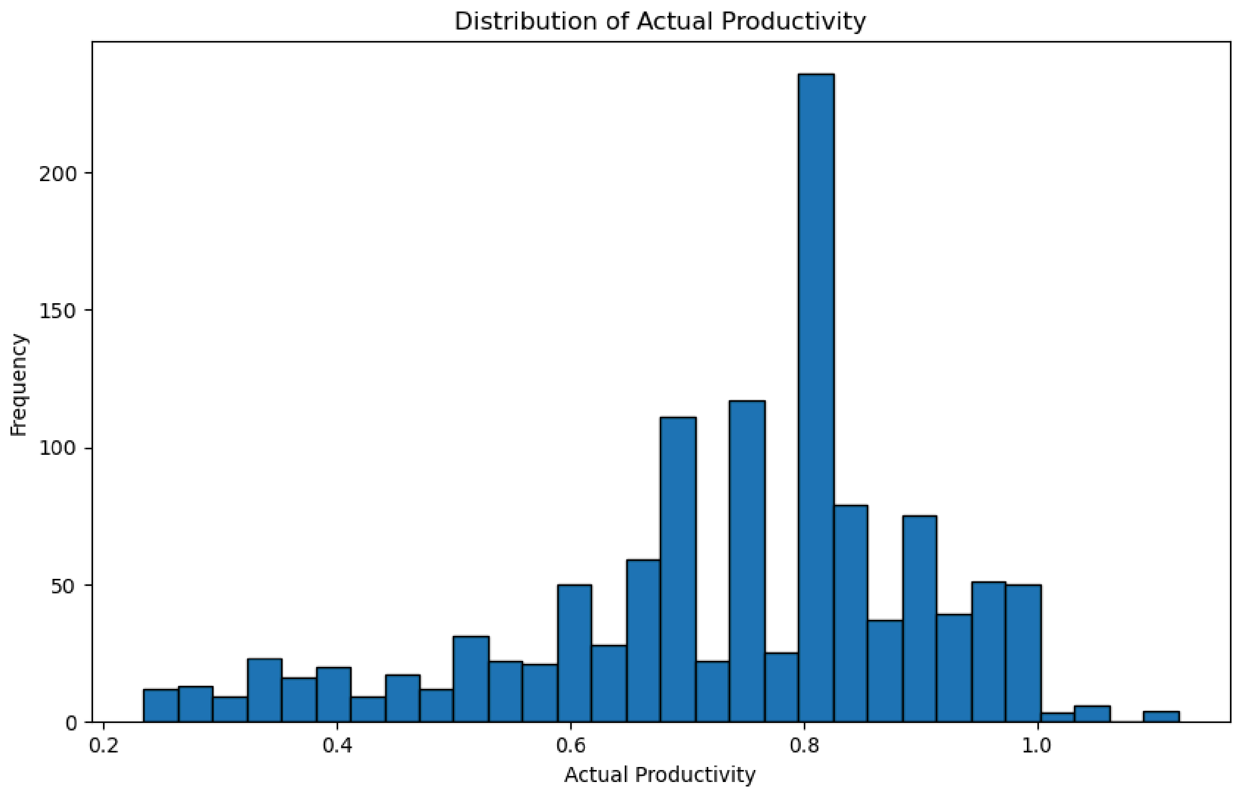

Figure 5. Most productivity values fall between 0.65 and 0.85, with a peak around 0.8.

3. <u>Methods</u>

3.1 Splitting

To evaluate the model effectively in the context of time-dependent data, I used Time Series Split as the splitting strategy. This approach ensures that the training set precedes the validation and test sets, avoiding data leakage and maintaining the temporal structure of the dataset. Specifically, I split the dataset into three segments:

- Training Set: From January 1, 2015, to January 18, 2015.
- Validation Set: From February 4, 2015, to February 23, 2015.
- Test Set: From February 23, 2015, to March 11, 2015.

3.2 Preprocessing

Data preprocessing was carefully designed to address the challenges posed by categorical and numerical features, as well as missing values. Categorical features such as department and team were processed using one-hot encoding, while ordinal features like day and quarter were encoded with ordinal encoding to preserve their inherent order. Numerical features, including targeted_productivity, smv, and over_time, were standardized using StandardScaler to ensure a consistent scale across all inputs. Missing data were handled using a reduced features method since it performed best with this dataset (Table 1).

Table 1. Ranked Methods (Best to Worst):

Reduced Features	0.0413
Multivariate Imputation	0.0429
XGB Direct Handling	0.0472

3.3 Hyperparameters and Cross Validation

The machine learning pipeline was tested with four algorithms, representing both linear and non-linear models. Ridge Regression and Lasso Regression were used as linear models. Ridge Regression addressed multicollinearity through L2 regularization. Lasso Regression introduced sparsity through L1 regularization, making it useful for feature selection. Non-linear models included Random Forest and K-Nearest Neighbors (KNN). Random Forest, a tree-based ensemble model, captured nonlinear interactions between features. KNN, a non-parametric model, captured local patterns by tuning n_neighbors (number of neighbors) and testing weights ("uniform" and "distance"). Table 2 shows the details of hyperparameters that each model tried with best parameters highlighted.

Table 2. ML Algorithms and Parameters:

ML Algorithms	**Parameters** ***Best Parameters**
Ridge Regression	alpha: [0.1, 1, 10, 100]
Lasso Regression	alpha: [0.01, 0.1, 1, 10]
Random Forest	n_estimators: [50, 100, 200] max_depth": [None, 10, 20] min_samples_split": [2, 5, 10]
K-Nearest Neighbors	n_neighbors: [3, 5, 7, 10] weights: ["uniform", "distance"]
Support Vector Regression	C: [0.1, 1, 10], kernel: ["linear", "rbf"] gamma: [0.01, 0.1, 1]

Model performance was evaluated using Mean Absolute Error (MAE) as the primary metric. MAE was chosen for its interpretability, as it represents errors in the same units as the target variable, actual_productivity. Additionally, it is robust against outliers compared to metrics like Mean Squared Error (MSE), making it well-suited for this context, where minimizing average errors is key to actionable insights.

To address uncertainties in evaluation, I incorporated strategies to quantify variability. For splitting variability, the Time Series Split ensured that temporal order was preserved, minimizing the influence of randomness in data segmentation. For non-deterministic models like Random

Forest, multiple runs with different random seeds were conducted to capture variability in performance metrics, ensuring reliable and robust evaluation.

4. Results

4.1 Baseline and Model Comparisons

The baseline model, which predicts the mean of the training data as the output, achieved a Mean Absolute Error (MAE) of 0.1203 and a Mean Squared Error (MSE) of 0.0287. This serves as a reference point for evaluating the performance of more complex models. Among the machine learning models tested, the Random Forest achieved the best predictive performance, with an MAE of 0.1192 and an MSE of 0.0293, slightly outperforming the baseline. In contrast, linear models like Ridge Regression and Lasso Regression demonstrated higher errors, with Ridge Regression having an MAE of 0.267 and Lasso Regression achieving an MAE of 0.163. The Support Vector Regression (SVR) performed the worst, with an MAE of 0.319 and an MSE of 1.531, indicating that it struggled to fit the dataset.

The Random Forest model's MAE is nearly one standard deviation (0.002) below the baseline, highlighting its ability to capture more complex patterns compared to the mean-based prediction. However, the small improvement over the baseline suggests that the dataset's inherent variability may limit the predictive power of more sophisticated models.

Table 3. Evaluation scores and standard deviation:

	MAE Mean	MAE Std	MSE Mean	MSE Std
Baseline	0.120316	0.000000	0.028788	0.000000
Ridge Regression	0.266801	0.000000	0.967681	0.000000
Lasso Regression	0.162905	0.000000	0.169143	0.000000
Random Forest	0.119261	0.001873	0.029357	0.000724
K-Nearest Neighbors	0.139149	0.000000	0.034680	0.000000
Support Vector Regression	0.318766	0.000000	1.530939	0.000000

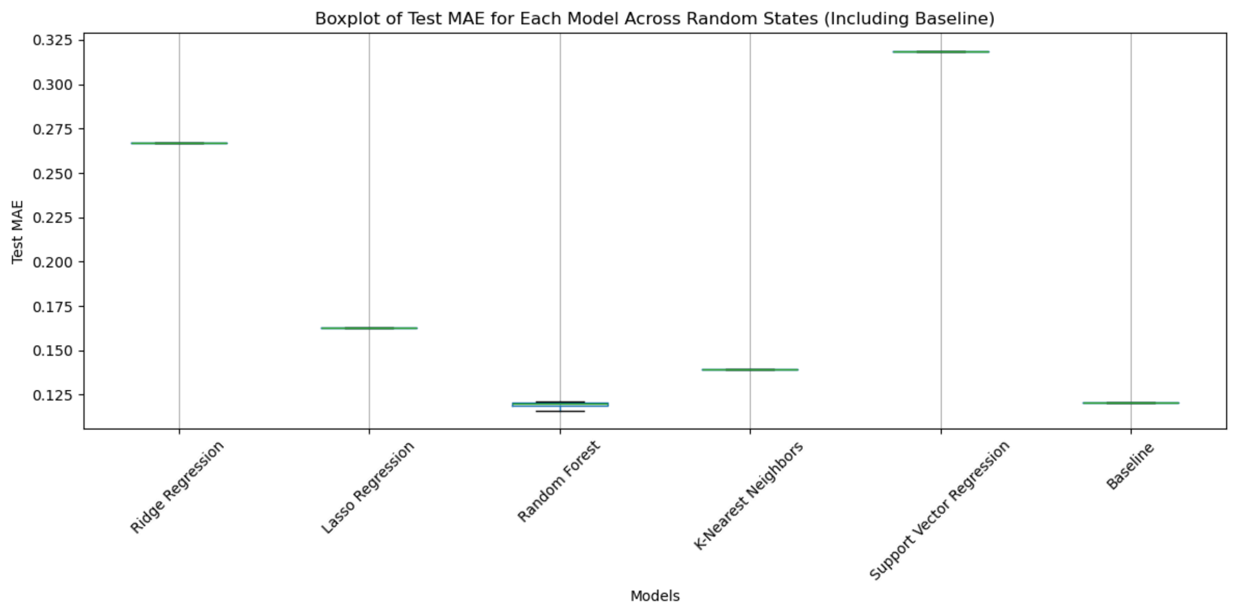

Figure 6. Comparison of MAE scores.

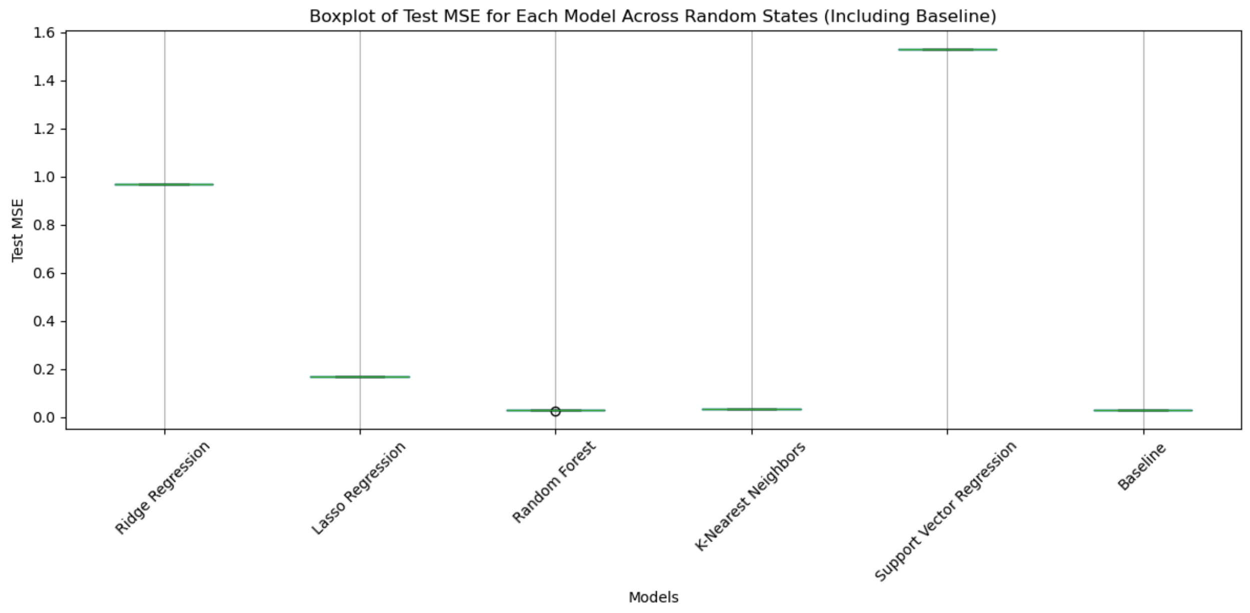

Figure 7. Comparison of MSE scores.

4.2 Feature Importance

To understand feature contributions, three methods of global feature importance were employed: permutation importance (Figure 8), gain-based importance (from XGBoost) (Figure 9), and SHAP values (Figure 10). Across all methods, targeted_productivity consistently emerged as the most important feature. This aligns with its strong correlation with the target variable identified in the exploratory data analysis. The features smv (Standard Minute Value) and over_time were also highly ranked, underscoring the influence of task complexity and extended work hours on productivity. Conversely, features such as is_weekend and team identifiers (e.g., team_7 and team_8) were among the least influential, suggesting that productivity is largely unaffected by these factors. The consistency of these results across models highlights the reliability of the identified key predictors.

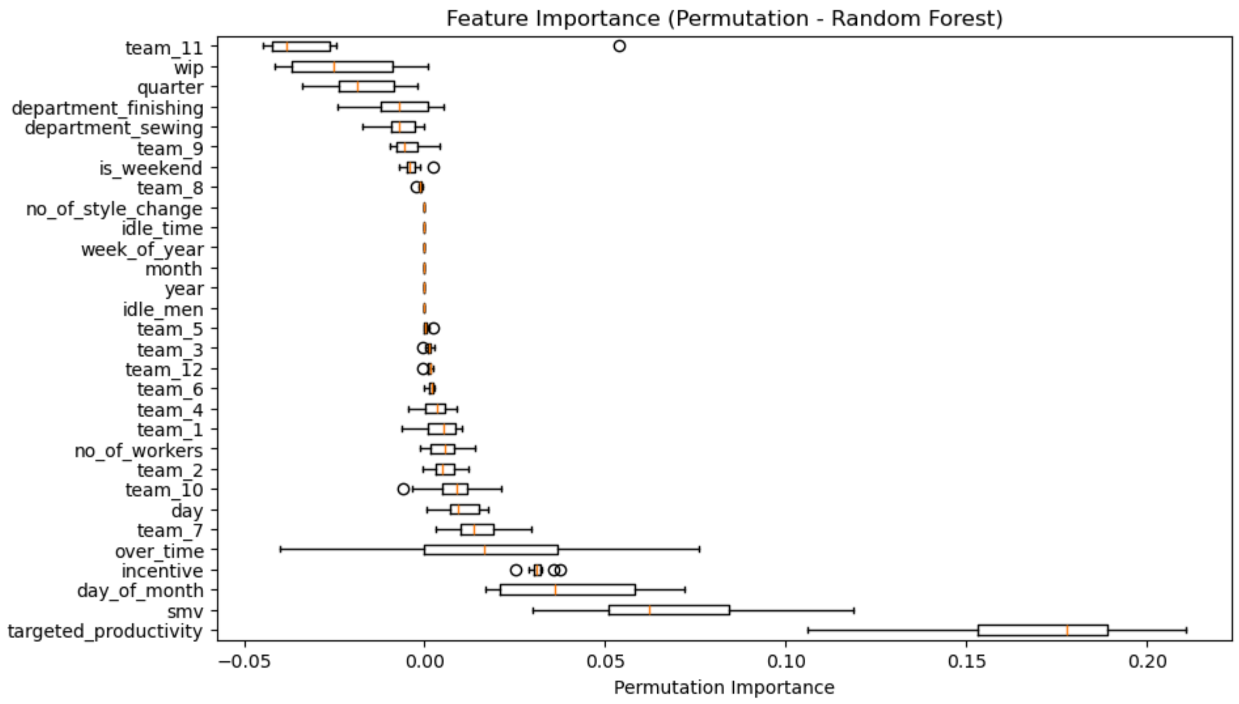

Figure 8. targeted_productivity was the most important feature, Least important features included is_weekend and team-specific variables (team_7, team_8).

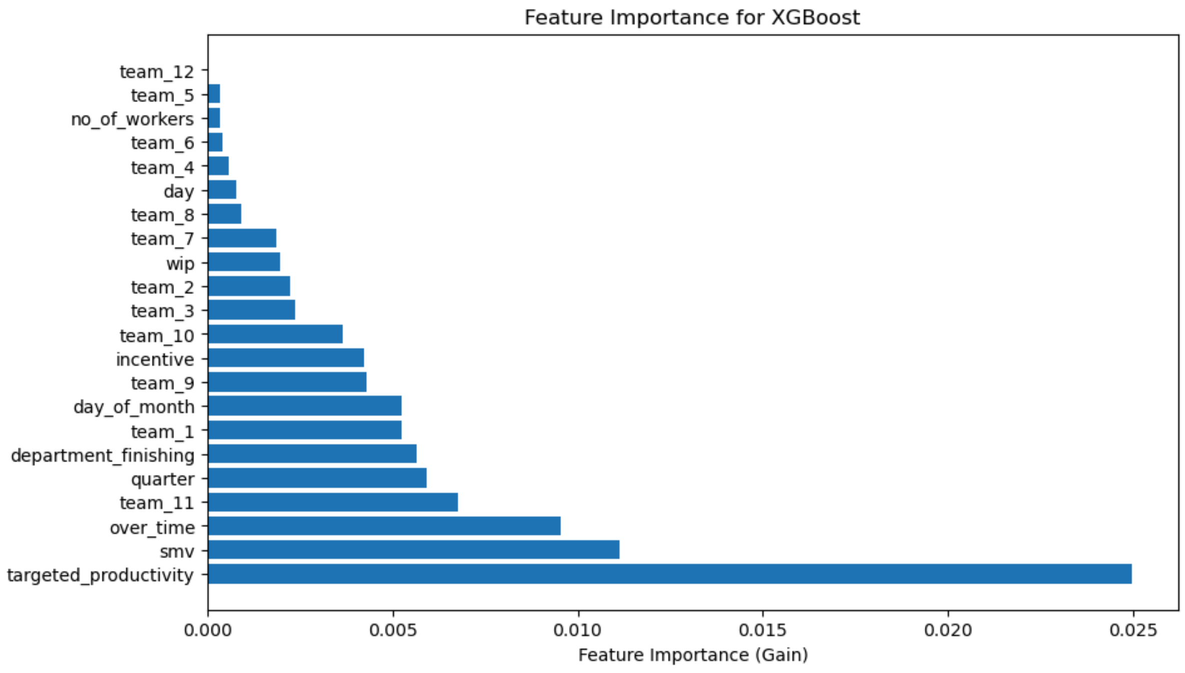

Figure 9. XGBoost highlighted targeted_productivity as the top feature, followed by smv and over_time.

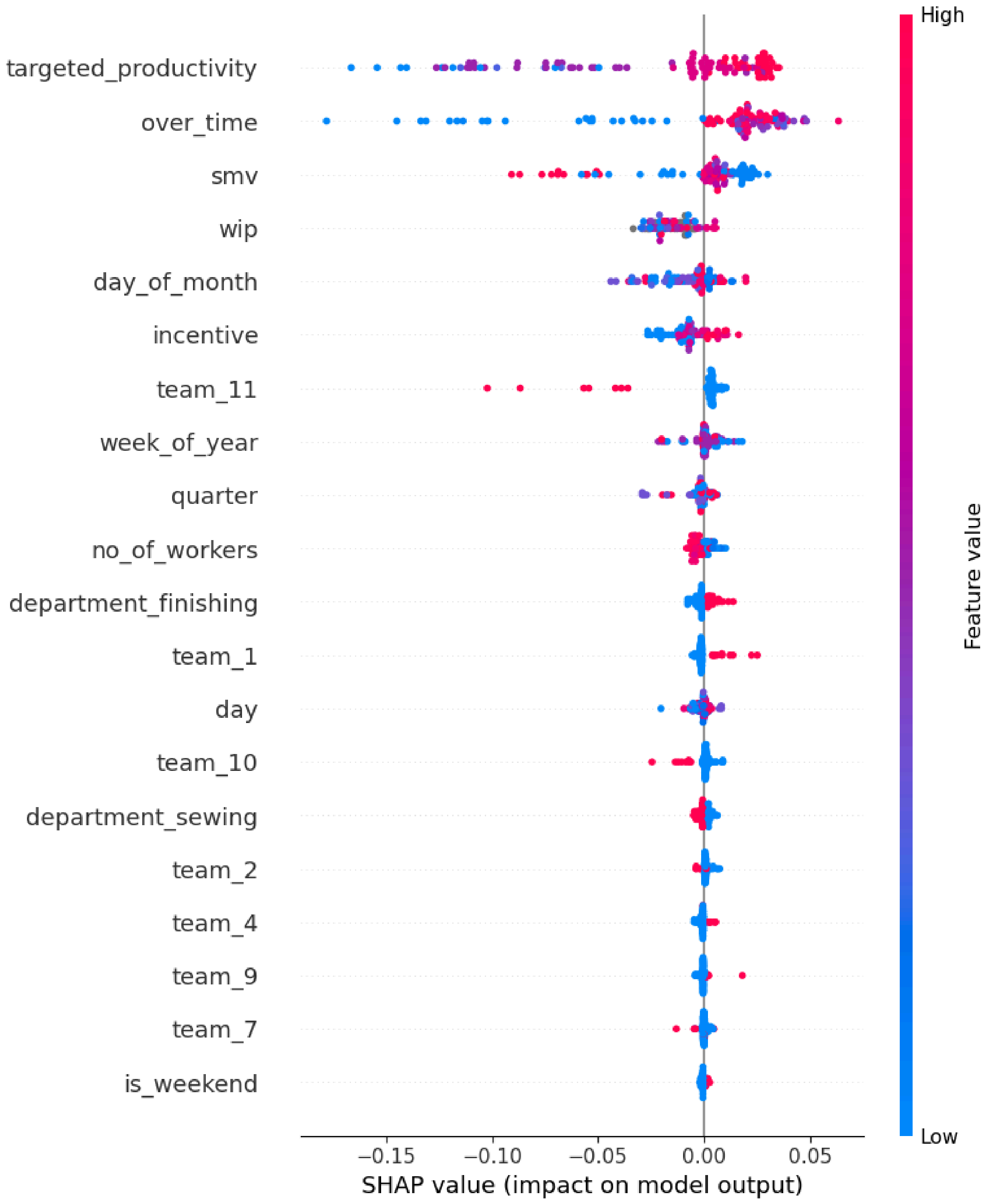

Figure 10. targeted_productivity had the largest positive impact on productivity predictions.

For local feature importance, SHAP analyses revealed how specific features influenced predictions for individual instances. For example, in index 37 (Figure 11), high over_time negatively impacted predictions, but this was mitigated by a high targeted_productivity value. In index 44 (Figure 12), smv played a significant role in reducing productivity, demonstrating how task complexity can vary in its impact on different cases.

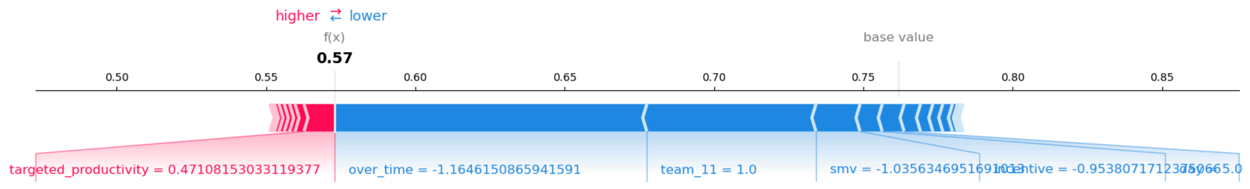

Figure 11. A low prediction was primarily influenced by high over_time.

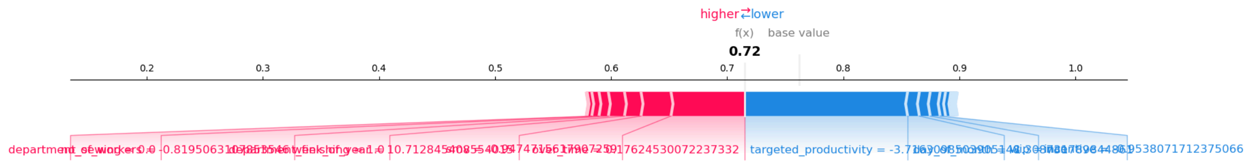

Figure 12. A high prediction was driven by department_finishing.

One surprising finding was the dual nature of over_time. While it is often associated with inefficiencies, the context of its occurrence can significantly alter its effect on productivity. Additionally, the minimal contribution of features such as is_weekend and some team-specific variables was unexpected, as team dynamics and work schedules were initially assumed to have a greater influence. These results emphasize the importance of task-specific variables over general contextual factors.

5. <u>Outlook</u>

The current model works, but there's definitely room to make it better. Random Forest performed the best, but its improvement over the baseline was minimal, which suggests we're not fully capturing the complexity of the problem. Some features, like is_weekend, didn't contribute much, and the lack of external factors like worker fatigue or machine downtime limits the model's ability. On top of that, the skewed productivity data makes it harder to predict unusual cases accurately.

To improve, we could create better features, like combining key variables or adding external data (e.g., worker profiles or operational metrics). Trying more hyperparameter options, especially for Random Forest and SVR, might uncover better results. Collecting more data, fixing the class imbalance, and testing advanced models like RNNs could also help. Adding cost-benefit analysis would make predictions more practical for real-world use.

These changes would make the model stronger and more insightful, helping it better predict and explain productivity patterns.

6. <u>References</u>

[1] ErningXu. “GitHub - ErningXu/Final_Report.” GitHub, 2024, https://github.com/ErningXu/Final_Report.git

[2] "Productivity Prediction of Garment Employees." UCI Machine Learning Repository, 2020, https://doi.org/10.24432/C51S6D.

.