

Project Report
Garment Worker Productivity Prediction

TEAM - 403

Team Members :

G.Varun
20BCE2365

B.Harish
20BCE7031

B. Deepak
20BES7019

V. Karthikeya Devara
20BES7035

1)INTRODUCTION

1.1 Overview

Garment Worker Productivity Forecast is a project designed to develop a forecasting model that can predict garment worker productivity based on a variety of factors. The production of garment workers plays an important role in the productivity and profitability of the garment industry.

This work involves using machine learning algorithms to analyze historical data and identify patterns and relationships between different devices and production workers. These entries can be work in progress (WIP), overtime, value minute value (SMV), quarter, day of the week, etc. It can contain attributes such as

By training a predictive model on historical data, it is possible to predict employee performance for new or unseen data.

This can provide insight and enable managers and stakeholders in the apparel industry to make decisions about business planning, budget allocation and production planning.

The purpose of the, Business Improvement Program is to increase productivity, reduce costs and improve the overall efficiency of the manufacturing process, traditional garments. By estimating the efficiency of employees, conflicts can be identified, productivity can be increased and work can be done more efficiently and production targets can be achieved.

This project has the potential to increase productivity, reduce waste and increase profitability in the apparel industry. It allows decision makers to make informed decisions, allocate resources efficiently and optimize staff utilization.

In summary, Labor Market Research used machine learning techniques to develop predictive models to predict workforce productivity in the apparel industry. Using the power of data and analytics, the program focuses on efficiency and enables businesses to reach new levels of productivity and efficiency.

1.2 Purpose

The aim of the Apparel Worker Efficiency Estimation project is to provide a better understanding and estimation of worker productivity in the apparel industry.

The project plan performs the following:

1)Resource Planning: The project estimates employee productivity, enabling managers and decision makers to plan and allocate resources efficiently. This includes determining the appropriate number of workers needed for a given job, planning the job, and making the best use of available resources.

2)Production Optimization: This project helps to improve the production process by identifying factors that affect employee productivity. By analyzing historical data and identifying patterns, the project can provide recommendation on how to improve operations, reduce bottlenecks, and improve overall results.

3) Performance Measurement: This project provides a quantitative measure of employee performance by estimating employee productivity. This can be used for performance reviews, identifying training needs, and rewarding high performers. It also allows managers to identify weak employees and take appropriate action to increase their productivity.

4)Cost reduction: Increased production can reduce costs in the clothing industry. The program will help reduce labor costs, optimize resources and reduce waste by predicting employee productivity. This can save clothing stores a lot of money.

5)Decision Support: This project provides decision makers with insights based on data analysis. It helps them make informed decisions about job planning, production scheduling, and resource allocation. These insights can help make better decisions and improve the overall efficiency of the manufacturing process.

In a nutshell, the goal of the Garment Worker Efficiency Estimation project is to use forecasting modeling techniques to improve resource planning, increase productivity, measure worker productivity, reduce costs, and give the apparel industry discretion. This work focuses on the best work and ultimately leads to the success and competitiveness of apparel companies.

2)LITERATURE SURVEY

2.1 Existing problem

In the context of garment workers' production forecasting, there are current challenges and many avenues explored. Below is a brief summary of the current problems and some solutions to solve them:

Main Problems:

- 1)Complexity and Variability: Apparel manufacturing involves complex and dynamic processes with multiple variables. Factors such as skill level, work environment, machine performance and job inequality are now challenging for accurate forecasting.
- 2)Data collection and performance: Obtaining comprehensive and reliable data on employee productivity can be difficult. Data collection methods such as log tracking or selfreporting can be misleading or biased, leading to inaccurate measurements.
- 3) Dynamic Working Conditions: Apparel production frequently changes working conditions, including different types of garments, production lines, or seasonal changes. Tuning forecasting models for such situations and integrating data in real time can be difficult.

Available methods and techniques:

- 1)Statistical methods: Statistical methods such as regression analysis are used to model the relationship between productivity and quantity type. This method aims to identify key determinants and develop mathematical models to predict product performance.
- 2)Time and Business Research: Time and Business Research Work to identify inefficiencies and suggest improvements. This approach can help identify specific tasks or activities that affect productivity.
- 3)Machine learning techniques: Machine learning algorithms, including linear regression, decision trees, random forests, support vector machines (SVM), and neural networks, have been used to build predictive models of garment worker productivity. This method uses historical data to identify patterns and relationships between different inputs and outputs.
- 4)Data Mining and Consensus Models: Data mining techniques such as clustering and cluster rule mining can be used to extract recommendations from data.

This process helps identify patterns and relationships that affect productivity.

5)Time series analysis: Time series analysis techniques can investigate time patterns and patterns of production workers. This process takes into account the nature of the data and can analyze seasonality, trends, etc. over time. can catch.

6)Internet of Things (IoT) and sensor technology: IoT and sensor technology can be used to collect realtime data about employee movement, technology, environment and other parameters. This information can be used to build predictive models and optimize employee performance.

It should be noted that each method has its own advantages and limitations, and the choice of method depends on the requirements and characteristics of the garment manufacturing process. Researchers and practitioners are exploring the combination of these methods to solve the problems associated with estimating garment workers.

2.2 Proposed solution

Garment workers' prediction generation solution involves using machine learning algorithms, specifically regression models, to create prediction models. The model aims to predict worker productivity from different inputs. Here is the outline of the plan:

1) Data collection and progress: Record historical data of production personnel, including relevant changes such as Work in Progress (WIP), Overall Workload, Transaction Minute Value (SMV),Quarterlyweeks and other measures.Preprocess data by processing missing values, encoding categorical variables, and measuring numeric variables if necessary.

2) Custom selection: Analyze data and select the most relevant information for production personnel. This step helps to reduce noise and increase the performance of the model.

3) Model training: divide the preprocessed data into training and test sets. Use a regression algorithm such as linear regression, random forest regression, gradient boost regression, or XGBoost regression to train the training model. These algorithms capture relationships between input variables and target Variables(efficiency mediators) to learn patterns and make predictions.

4) Model Evaluation: Evaluate the effectiveness of the training model using appropriate metrics such as Mean Squared Error (MSE) and Mean Absolute Error

(MAE). These metrics evaluate the accuracy of model predictions compared to actual production results in the measurement process.

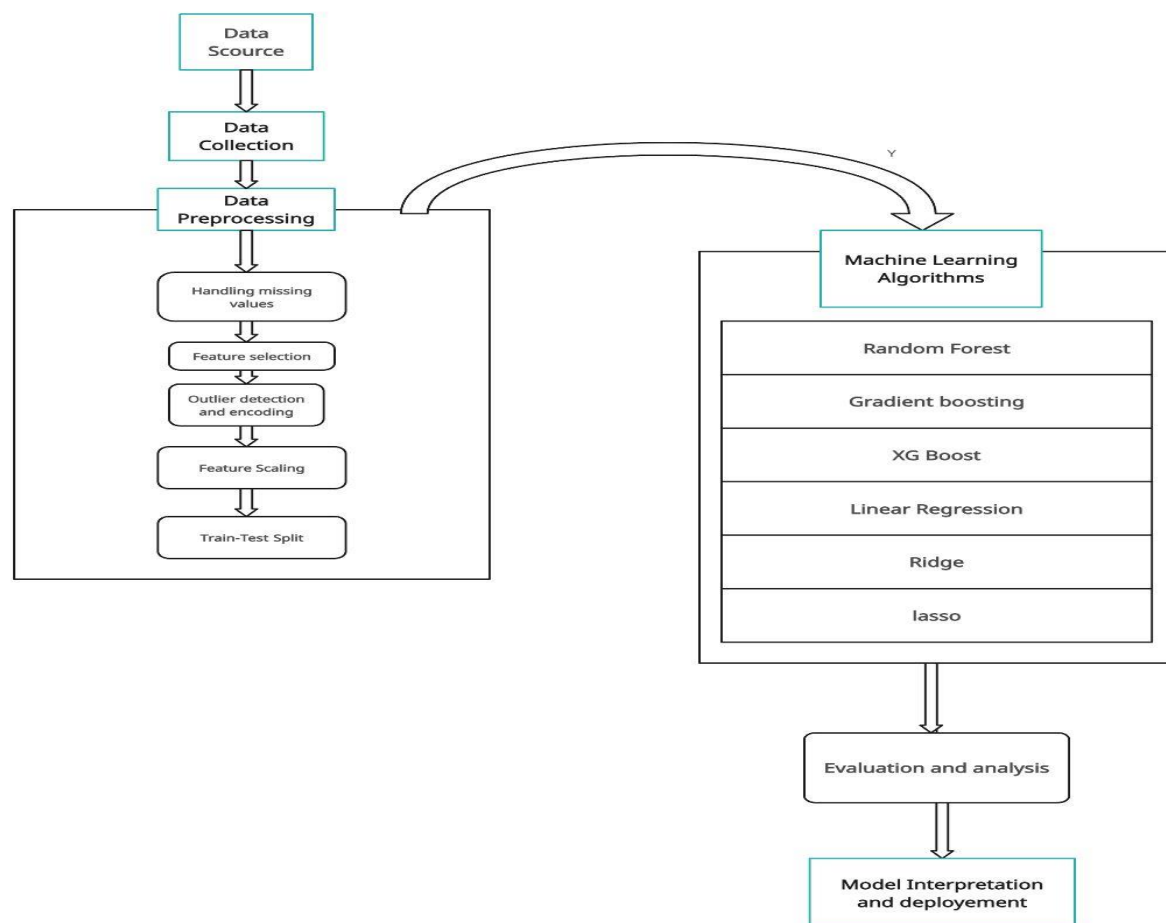
5) Model selection and optimization: Compare the performance of different models and select the model with the best performance according to the evaluation criteria. Finetune the selected model to improve its performance by adjusting the hyperparameters.

6) Deploy the Model: Once the best model is selected and optimized, it can be saved and deployed for use in production. The model can be integrated into a web application using frameworks like Flask, where users can input relevant variables, and the model predicts the worker productivity.

It is important to note that the regression algorithm selection and prioritization process will vary depending on the specific characteristics of the data and the needs of the design process. The solution provides a general framework for developing a forecasting model to forecast the clothing workforce.

3 THEORITICAL ANALYSIS

3.1 Block diagram



3.2 Hardware / Software designing

The hardware and software requirements for the Garment Worker Productivity Forecasting Project are as follows:

Hardware Requirements:

1. Computer or Server: A computer or server with sufficient processing power and memory to manage data preforms, training models, and outputs.

Software Requirements:

1) Python: The project will be implemented using the Python programming language.

2) Integrated Development Environment (IDE): Coding and development requires an IDE such as Jupyter Notebook, PyCharm, or Anaconda.

3) Python libraries: Use the pip or conda package manager to install the appropriate Python libraries. Required libraries include:

- Pandas: For data management and prioritization.
- NumPy: For arithmetic.
- Scikit-Learn: For machine learning algorithms and model evaluation.
- Matplotlib: For data visualization and plotting.
- Bottle: For developing web applications.
- XGBoost: For using XGBoost regression models (if selected).
- Joblib: To save and load training models.
- Additional libraries on special request.

Also requires internet connection to download files, linked libraries and files (if required).

Note that hardware requirements may vary based on file size, model complexity and hardware components. It is recommended to have a computer or server with enough power, enough RAM and storage capacity to perform well in data processing and modeling tasks.

4 EXPERIMENTAL INVESTIGATIONS

During the experimental research on garment productivity prediction, many observations and observations have been made to create a good predictive model. The following are some of the key research done during the development of the solution:

1) Data Analysis: Analyze the data to understand the distribution and characteristics of the difference. Check metrics like mean, standard deviation, minimum and maximum to understand your data.

2) Data Visualization: Use a variety of data visualizations to explore the relationship between target variables (employee productivity) and input data. Scatter

charts, histograms, and 3D charts are used to identify patterns or relationships in data.

3) Special choice: The effect of different strategies on the production staff has been examined. Use correlation analysis, values for random forest models, and domain information to select the most important features for the prediction model.

4) Model training and evaluation: train and evaluate multiple regression models, including linear regression, random forest regression, gradient boost regression and XGBoost regression. Models are evaluated using metrics such as the square of mean error (MSE) and mean error (MAE) to determine how well they predict workers.

5) Model Comparison: Compare the performance of different regression models to determine which model performs best. These models are evaluated for their ability to reduce forecast error and provide good forecast performance.

6) Hyperparameter tuning: Use hyperparameter tuning techniques such as grid search or random search to improve the performance of selected models. Different combinations of hyperparameters were tried to find the best configuration for the model.

7) Model Delivery and Testing: Use sample selection and optimization in web applications using Flask. The web application is tested to ensure it can predict employee productivity based on user input.

The aim of these surveys is to create a powerful and accurate forecasting model that can effectively predict the productivity of garment workers. These analyzes and searches help to understand data, select relevant features, compare models and improve their performance, ultimately leading to improved print job performance.

5 ADVANTAGES & DISADVANTAGES

Advantages of the Proposed Solution:

1) Better production forecasting: The proposed solution uses machine learning algorithms to predict the production of garment workers. This allows for better planning and allocation of resources, resulting in more accurate forecasting than traditional methods.

2) Automation and Efficiency: The solution will automate the productivity estimation process, reducing the need for manual calculations and subjective estimates. This increases efficiency and saves time for managers and decision makers.

3) Data Decision Making: Solutions use historical data to make predictions. This enables data-driven decision making, allowing managers to rely on objective insights rather than relying solely on intuition or experience.

4) Scalability: Solutions can scale to handle large datasets and accommodate additional features or changes as needed. This scalability provides flexibility to adapt to the needs of the business.

5) Model Evaluation and Selection: This solution involves evaluating multiple regression models to select the best performing model. This ensures that the selected model provides the most accurate estimate for garment manufacturing workers.

Disadvantages of the Proposed Solution:

1) Data Availability and Quality: The accuracy of the forecast is highly dependent on the availability and quality of historical data. Missing, inconsistent or biased data can affect the performance and reliability of prediction models.

2) Model complexity: Machine learning models such as random forest or gradient boosting can be complex and require computational resources for training and inference. This can be a problem for organizations without computing power or limited technical knowledge.

3) Interpretability: Some machine learning models, such as black box models, may not be interpretable. Understanding the principles or variables that affect production forecasting can be difficult, hindering clarity and decision making.

4) Model Maintenance and Updates: Solutions need regular maintenance and updates to ensure model accuracy and accuracy. New information must be compiled and models must be reworked from time to time to adapt to changing models or standards.

5) Dependency on Input Variables: The accuracy of productivity estimates depends on the availability and accuracy of input variables such as WIP, overtime, SMV, and others. Failure to measure or document these changes will affect the reliability of the estimates.

It is important to consider these advantages and disadvantages when implementing solutions and solving problems to ensure their effectiveness and usability.

6 APPLICATIONS

The needs of garment workers create predictive solutions that can be used in

many industries and where the maintenance and optimization of workers is important. Some of the applications of this solution are:

1)Garment Manufacturing: This solution can be used specifically in the garment manufacturing industry to predict and increase employee productivity. It helps managers and production planners make decisions about resource allocation, job scheduling, and production schedules.

2)Business Management: This solution is suitable for construction, assembly line production, logistics, etc. It can be used in industries that rely heavily on manual labor, such as It provides better performance management, performance and performance evaluation by predicting the performance of employees.

3) Capacity Planning: Organizations that need capacity estimation can benefit from this solution. By estimating employee productivity, they can predict output and plan capacity needs accordingly, ensuring efficient use of resources.

4) Performance Monitoring: The solution can be used to monitor and track the performance of an individual employee or team over time. It provides insight into factors affecting productivity, allowing organizations to identify areas of improvement and implement training plans or process improvements.

5)Operational Efficiency: Businesses that perform missioncritical tasks such as retail, retail, and ecommerce can use this solution to improve energy efficiency and overall performance.It helps identify bottlenecks, inefficiencies and areas for improvement in business processes.

6) Project Management: In projectbased industries such as construction, engineering, and software development, this solution helps project managers anticipate and manage the people working efficiently throughout the project lifecycle. Assists with resource planning, operations, and scheduling.

7)Decision Support: Solutions can provide decision support for managers and executives in many industries. It provides them with useful forecasts by enabling them to make datadriven decisions regarding performance management, performance improvement and resource allocation.

By implementing these solutions in these and other areas, organizations can increase productivity, increase operational efficiency, and make informed decisions based on apparel workers' forecasts.

7 CONCLUSIONS

In a nutshell, the Business Department's goal is to create solutions for apparel

workers to predict and predict their products using predetermined standard machine learning. The project included several key steps, including creating a web application for preliminary data, model training, evaluation and forecast time. Through analysis and analysis of the dataset, it was found that variables such as Work In Progress, overtime hours, SMV, quarter, and day of the week were associated with the impact on production workers. Clean data, process missing values, and perform coding to prepare data for modelling.

Test and compare multiple regression models, including linear regression, random forest regression, gradient boost regression, and XGBoost regression. The best model is chosen based on statistical measures such as the square of the mean error (MSE) and the mean error (MAE).

The solution has many benefits such as improved forecasting, automation, decision making, capacity building and model evaluation. However, it also has some limitations such as data availability and quality, model complexity, interpretation and maintenance.

A web application developed by provides a userfriendly interface for users to enter relevant changes and get time estimates for garment workers. This enables managers and decision makers to make informed decisions, improve resource allocation and increase efficiency.

Overall, the project demonstrates the potential of machine learning to predict and predict garment manufacturing workers. The findings highlight the importance of accurate data, sample selection and continuous monitoring to ensure the reliability and effectiveness of solutions. Future enhancements will include additional functionality, improved translation, and addressing limitation found in the process.

8 FUTURE SCOPE

The Department of Business has many areas for future development and advancement. Here are some possible suggestions for the future:

- 1) Provide realtime data: Currently, solutions use historical data for product forecasting. In the future, the integration of realtime data, such as IoT devices or sensors, can further predict and update. This will allow for a more efficient and effective estimation.
- 2) Improvement of the model: continuous improvement of the prediction model can be explored. Techniques such as hyperparameter tuning and ensemble learning can be used to improve the performance of the model and further

increase the accuracy of the prediction performance.

3) Feature Engineering: It can increase the predictive power of the model by adding important features or creating new engineering techniques.

Discovering the differences between bringing different or specific experiences together can provide a deeper understanding of what influences garment workers.

4) Model Interpretability: Improved interpretability of machine learning models can help stakeholders understand what leads to good results. Techniques such as factor analysis, pattern identification or visualization can provide insight into key drivers of productivity and support decision making.

5) Integration with advanced analytics: Integration of product forecasting solutions with advanced analytics systems such as predictive maintenance or demand forecasting towers can get the job done. This integration can enable organizations to make more informed decisions and improve resource allocation in all aspects of the business.

6) Collaboration and Feedback Mechanisms: Using feedback strategies to adjust models for real production data collection integration and continuous improvement li. Collaborating with experts and staff can provide valuable insights and unparalleled expertise to improve forecast accuracy and precision.

7) Scalability and deployment: The solution can be adapted to a larger database and more production sites or facilities. Creating a stable and productive environment enables connectivity and utilization across multiple locations and businesses.

8) Integration with Human Resource Management Systems: Integration of product forecasting with existing human resource management systems can simplify business planning, performance measurement and resource allocation processes. This integration supports seamless data management and provides productivity control.

Human resource management systems can improve by exploring these future developments and can be used in apparel industry etc. can provide greater clarity, efficiency and effectiveness for productivity.

9 REFERENCES

1. H. H. Sabuj, N. S. Nuha, P. R. Gomes, A. Lameesa and M. A. Alam, "Interpretable Garment Workers' Productivity Prediction in Bangladesh Using Machine Learning Algorithms and Explainable AI," 2022 25th International Conference on Computer and Information Technology (ICCIT), Cox's Bazar, Bangladesh, 2022, pp. 236-241, doi: 10.1109/ICCIT57492.2022.10054863.
2. <https://stackoverflow.com/questions/18296755/python-max-function-using-key-and-lambda-expression>.
3. N. S. S. V. S. Rao, S. J. J. Thangaraj and V. S. Kumari, "Flight Ticket Prediction Using Gradient Boosting Regressor Compared With Linear Regression," 2023 Eighth International Conference on Science Technology Engineering and Mathematics (ICONSTEM), Chennai, India, 2023, pp. 1-6, doi: 10.1109/ICONSTEM56934.2023.10142428.

10 APPENDIX :

Source Code:

Py code:

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# In[2]:

df = pd.read_csv("productivity.csv")
df.head()
```

```
# Cleaning the data

# In[3]:

df = df.drop(["date", "department", "idle_time", "idle_men"], axis=1)

# In[4]:

df=df.dropna()

# In[5]:

df

# In[6]:

df

# In[7]:

df.describe()

# In[8]:

plt.figure(figsize=(18, 16))
plt.hist(df, bins=2000, edgecolor="cyan")
plt.xlabel("Targeted Productivity")
plt.ylabel("Frequency")
plt.title("Distribution of Targeted Productivity")
plt.show()

# In[9]:

plt.figure(figsize=(10, 8))
df.boxplot(column="targeted_productivity", by="day", grid=False)
plt.xlabel("Day of the Week")
```

```

plt.ylabel("Targeted Productivity")
plt.title("Targeted Productivity by Day of the Week")
plt.show()

# In[10]:

import seaborn as sns
sns.set(style="ticks")
sns.pairplot(df, vars=["targeted_productivity", "smv", "wip", "over_time"])
plt.title("Scatter Matrix")
plt.show()

# In[11]:

correlation = df.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation, annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap")
plt.show()

# In[12]:

import plotly.express as px
fig = px.line(df, x='over_time', y='targeted_productivity', title='Targeted Productivity over Time')
fig.show()

# In[13]:

import plotly.graph_objects as go

fig = go.Figure(data=[go.Scatter3d(
    x=df['wip'],
    y=df['over_time'],
    z=df['targeted_productivity'],
    mode='markers',
    marker=dict(
        size=7,
        color=df['targeted_productivity'],
        colorscale='Viridis',
        opacity=0.8
    )
)])

```

```

    ))

fig.update_layout(scene=dict(
    xaxis_title='WIP',
    yaxis_title='Overtime',
    zaxis_title='Targeted Productivity'
))

fig.show()

# In[14]:

day_avg_productivity = df.groupby("day")["targeted_productivity"].mean()

fig = px.bar(day_avg_productivity, x=day_avg_productivity.index,
y=day_avg_productivity.values,
             labels={'x': 'Day of the Week', 'y': 'Average Targeted
Productivity'},
             title='Day of the Week vs. Average Targeted Productivity')

fig.show()

# In[15]:

fig = px.violin(df, x="quarter", y="targeted_productivity", box=True,
points="all", title="Targeted Productivity by Quarter")
fig.show()

# In[16]:

smv = df["smv"]

plt.figure(figsize=(8, 6))
plt.scatter(smv, df["targeted_productivity"], alpha=0.5)
plt.xlabel("SMV")
plt.ylabel("Targeted Productivity")
plt.title("Targeted Productivity vs. SMV")
plt.show()

# In[17]:

wip = df["wip"]

```



```

overtime = df["over_time"]

fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection="3d")
ax.scatter(wip, overtime, df["targeted_productivity"],
c=df["targeted_productivity"], cmap="coolwarm", alpha=0.8)
ax.set_xlabel("WIP")
ax.set_ylabel("Overtime")
ax.set_zlabel("Targeted Productivity")
ax.set_title("Multivariate Analysis: WIP, Overtime, and Targeted
Productivity")
plt.show()

# In[18]:

df_encoded = pd.get_dummies(df, columns=["quarter", "day"])

X = df_encoded.drop("targeted_productivity", axis=1)
y = df_encoded["targeted_productivity"]

# In[19]:

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# In[33]:

linear_reg = LinearRegression()
linear_reg.fit(X_train, y_train)
linear_pred = linear_reg.predict(X_test)
linear_mse = mean_squared_error(y_test, linear_pred)
linear_mae = mean_absolute_error(y_test, linear_pred)
linear_r2 = r2_score(y_test, linear_pred)

# In[21]:

rf_reg = RandomForestRegressor(random_state=42)
rf_reg.fit(X_train, y_train)
rf_pred = rf_reg.predict(X_test)
rf_mse = mean_squared_error(y_test, rf_pred)
rf_mae = mean_absolute_error(y_test, rf_pred)
rf_r2 = r2_score(y_test, rf_pred)

```

```
# In[22]:
```

```
gb_reg = GradientBoostingRegressor(random_state=42)
gb_reg.fit(X_train, y_train)
gb_pred = gb_reg.predict(X_test)
gb_mse = mean_squared_error(y_test, gb_pred)
gb_mae = mean_absolute_error(y_test, gb_pred)
gb_r2 = r2_score(y_test, gb_pred)
```

```
# In[23]:
```

```
import xgboost as xgb
```

```
# In[24]:
```

```
xgb_reg = xgb.XGBRegressor(random_state=42)
xgb_reg.fit(X_train, y_train)
xgb_pred = xgb_reg.predict(X_test)
xgb_mse = mean_squared_error(y_test, xgb_pred)
xgb_mae = mean_absolute_error(y_test, xgb_pred)
xgb_r2 = r2_score(y_test, xgb_pred)
```

```
# In[25]:
```

```
ridge_reg = Ridge(random_state=42)
ridge_reg.fit(X_train, y_train)
ridge_pred = ridge_reg.predict(X_test)
ridge_mse = mean_squared_error(y_test, ridge_pred)
ridge_mae = mean_absolute_error(y_test, ridge_pred)
ridge_r2 = r2_score(y_test, ridge_pred)
```

```
# In[26]:
```

```
lasso_reg = Lasso(random_state=42)
lasso_reg.fit(X_train, y_train)
lasso_pred = lasso_reg.predict(X_test)
lasso_mse = mean_squared_error(y_test, lasso_pred)
lasso_mae = mean_absolute_error(y_test, lasso_pred)
lasso_r2 = r2_score(y_test, lasso_pred)
```

```
# In[ ]:
```

```
# In[34]:
```

```
models = {  
    "Linear Regression": linear_reg,  
    "Random Forest Regression": rf_reg,  
    "Gradient Boosting Regression": gb_reg,  
    "XGBoost Regression": xgb_reg,  
    "Ridge Regression": ridge_reg,  
    "Lasso Regression": lasso_reg  
}
```

```
# In[35]:
```

```
for model_name, model in models.items():  
    if isinstance(model, tuple):  
        model = model[0]  
    y_pred = model.predict(X_test)  
    mse = mean_squared_error(y_test, y_pred)  
    r2 = r2_score(y_test, y_pred)  
  
    print(model_name + ":")  
    print("Mean Squared Error:", mse)  
    print("R2 Score:", r2)  
    print()
```

```
# In[39]:
```

```
best_model = max(models, key=lambda x: r2_score(y_test,  
models[x].predict(X_test)))  
print("Best Model:", best_model)
```

```
# In[40]:
```

```
pip install joblib
```

```

# In[41]:

import joblib

# In[44]:

best_model_name = max(models, key=lambda x: r2_score(y_test,
models[x].predict(X_test)))
best_model = models[best_model_name]

joblib.dump(best_model, 'best_model.pkl')

```

Flask_app.py:

```

from flask import Flask, render_template, request
import joblib
import pandas as pd

app = Flask(__name__)
model = joblib.load('best_model.pkl')

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():
    if request.method == 'POST':
        smv = float(request.form['smv'])
        wip = int(request.form['wip'])
        overtime = int(request.form['overtime'])
        quarter = request.form['quarter']
        day = request.form['day']

        user_data = pd.DataFrame({
            'smv': [smv],
            'wip': [wip],
            'over_time': [overtime],
            'quarter': [quarter],
            'day': [day]
        })

        user_data_encoded = pd.get_dummies(user_data, columns=['quarter',
'day'])

```

```

X_columns = model.estimators_[0].feature_importances_
X = user_data_encoded.reindex(columns=X_columns, fill_value=0)

prediction = model.predict(X)[0]

return render_template('result.html', prediction=prediction)

if __name__ == '__main__':
    app.run(debug=True)

```

Style.css:

```

body {
    font-family: Arial, sans-serif;
    background-color: #f4f4f4;
    --text-color: #333;
    --background-color: #fff;
    --form-border-color: #ccc;
    --form-border-focus-color: #555;
    --button-background-color: #333;
    --button-color: #fff;
}

.container {
    max-width: 400px;
    margin: 0 auto;
    padding: 20px;
    background-color: var(--background-color);
    border-radius: 5px;
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
}

h1 {
    text-align: center;
    color: var(--text-color);
}

form {
    margin-top: 20px;
}

label {
    display: block;
    margin-bottom: 10px;
    color: var(--text-color);
}

```

```

input[type="number"],
select {
  width: 100%;
  padding: 10px;
  margin-bottom: 15px;
  border: 1px solid var(--form-border-color);
  border-radius: 4px;
  transition: border-color 0.3s ease;
  background-color: var(--background-color);
  color: var(--text-color);
}

input[type="number"]:focus,
select:focus {
  border-color: var(--form-border-focus-color);
}

input[type="submit"] {
  width: 100%;
  padding: 12px;
  background-color: var(--button-background-color);
  color: var(--button-color);
  border: none;
  border-radius: 4px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

input[type="submit"]:hover {
  background-color: #555;
}

.dark-mode {
  --text-color: #fff;
  --background-color: #333;
  --form-border-color: #999;
  --form-border-focus-color: #fff;
  --button-background-color: #555;
  --button-color: #fff;
}

.dark-mode .container {
  box-shadow: 0 2px 5px rgba(255, 255, 255, 0.1);
}

```

Result.css:

```
/* result.css */

body {
  font-family: Arial, sans-serif;
  background-color: #f8f8f8;
  color: #333;
  transition: background-color 0.3s ease, color 0.3s ease;
}

.dark-mode body {
  background-color: #333;
  color: #f8f8f8;
}

h1 {
  text-align: center;
  color: inherit;
}

h3 {
  margin-bottom: 10px;
  color: inherit;
}

.result-container {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}

.result-box {
  padding: 20px;
  background-color: inherit;
  border-radius: 5px;
  box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
  transition: box-shadow 0.3s ease;
}

.dark-mode .result-box {
  background-color: #222;
  box-shadow: 0 2px 5px rgba(255, 255, 255, 0.1);
}

.result-box:hover {
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
}
```

```
.result h3 {
  margin-bottom: 10px;
  color: inherit;
}

.result-value {
  font-size: 24px;
  font-weight: bold;
  text-align: center;
  color: inherit;
}

.dark-mode .result-value {
  color: #f8f8f8;
}

.dark-mode input[type="checkbox"] {
  transform: rotate(180deg);
}

.dark-mode .toggle-label:before {
  background-color: #333;
}

.toggle-container {
  display: flex;
  justify-content: center;
  margin-top: 20px;
}

.toggle-label {
  position: relative;
  display: inline-block;
  width: 50px;
  height: 26px;
  background-color: #ccc;
  border-radius: 13px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.toggle-label:before {
  content: "";
  position: absolute;
  top: 1px;
  left: 1px;
  width: 24px;
  height: 24px;
```



```

background-color: #f8f8f8;
border-radius: 50%;
transition: transform 0.3s ease, background-color 0.3s ease;
}

.toggle-checkbox:checked + .toggle-label {
background-color: #4caf50;
}

.toggle-checkbox:checked + .toggle-label:before {
transform: translateX(24px);
background-color: #fff;
}

```

Index.html:

```

<!DOCTYPE html>
<html>
<head>
<title>Garment Productivity Prediction</title>
<link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
<h1>Garment Productivity Prediction</h1>
<form action="{{ url_for('predict') }}" method="POST">
<label for="smv">SMV:</label>
<input type="number" step="0.01" name="smv" required><br><br>
<label for="wip">WIP:</label>
<input type="number" name="wip" required><br><br>
<label for="overtime">Overtime:</label>
<input type="number" name="overtime" required><br><br>
<label for="quarter">Quarter:</label>
<select name="quarter" required>
<option value="Q1">Q1</option>
<option value="Q2">Q2</option>
<option value="Q3">Q3</option>
<option value="Q4">Q4</option>
</select><br><br>
<label for="day">Day:</label>
<select name="day" required>
<option value="Saturday">Saturday</option>
<option value="Sunday">Sunday</option>

```

```

<option value="Monday">Monday</option>
<option value="Tuesday">Tuesday</option>
<option value="Wednesday">Wednesday</option>
<option value="Thursday">Thursday</option>
<option value="Friday">Friday</option>
</select><br><br>
<div class="dark-mode-toggle">
<input type="checkbox" id="darkModeToggle">
<label for="darkModeToggle">Dark Mode</label>
</div>
<input type="submit" value="Predict">
</form>
<script>
const darkModeToggle = document.getElementById('darkModeToggle');
darkModeToggle.addEventListener('change', () => {
document.body.classList.toggle('dark-mode');
});
</script>
</body>
</html>

```

Result.html:

```

<!DOCTYPE html>
<html>
<head>
<title>Garment Productivity Prediction - Result</title>
<link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='result.css') }}">
</head>
<body>
<div class="toggle-container">
<input type="checkbox" class="toggle-checkbox" id="dark-mode-toggle">
<label class="toggle-label" for="dark-mode-toggle"></label>
</div>
<div class="result-container">
<div class="result-box">
<h3>Predicted Productivity:</h3>
<p class="result-value">{{ prediction }}</p>
</div>
</div>
<script>

```

```

const darkModeToggle = document.getElementById('dark-mode-toggle');
const body = document.body;

// Check the saved user preference for dark mode
const darkModePreferred = localStorage.getItem('darkMode');

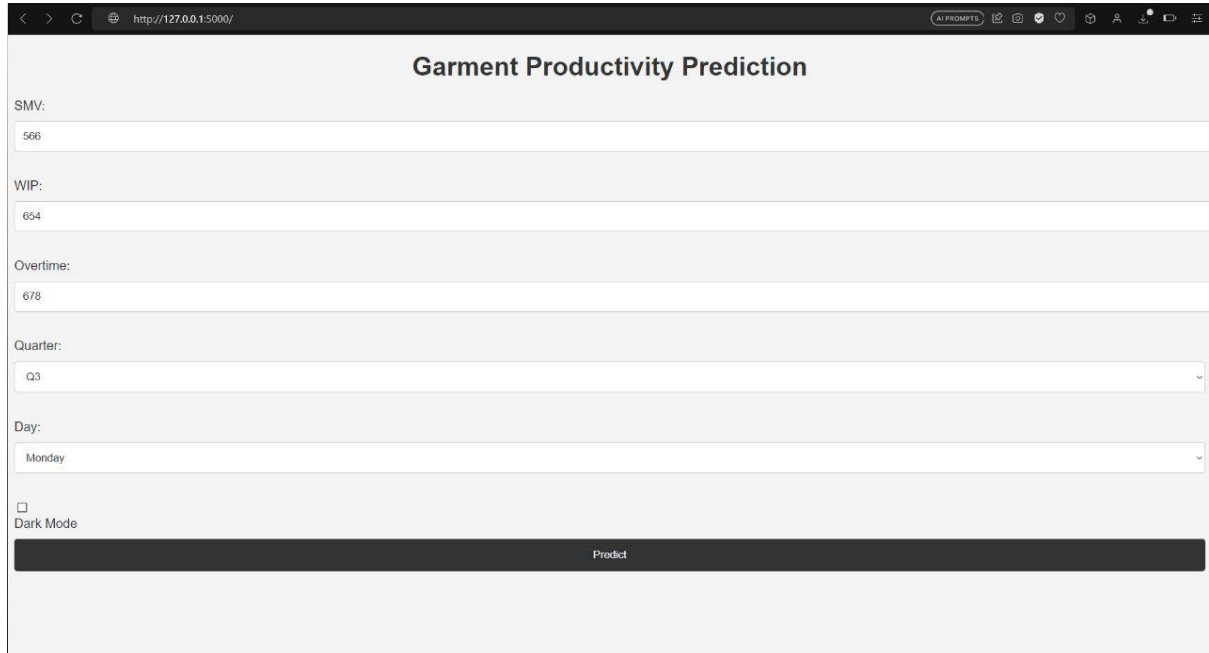
// Set the initial state based on the user preference
if (darkModePreferred === 'true') {
  darkModeToggle.checked = true;
  body.classList.add('dark-mode');
}

// Toggle dark mode when the checkbox is clicked
darkModeToggle.addEventListener('change', () => {
  body.classList.toggle('dark-mode');

  // Save the user preference for dark mode
  localStorage.setItem('darkMode', darkModeToggle.checked);
});
</script>
</body>
</html>

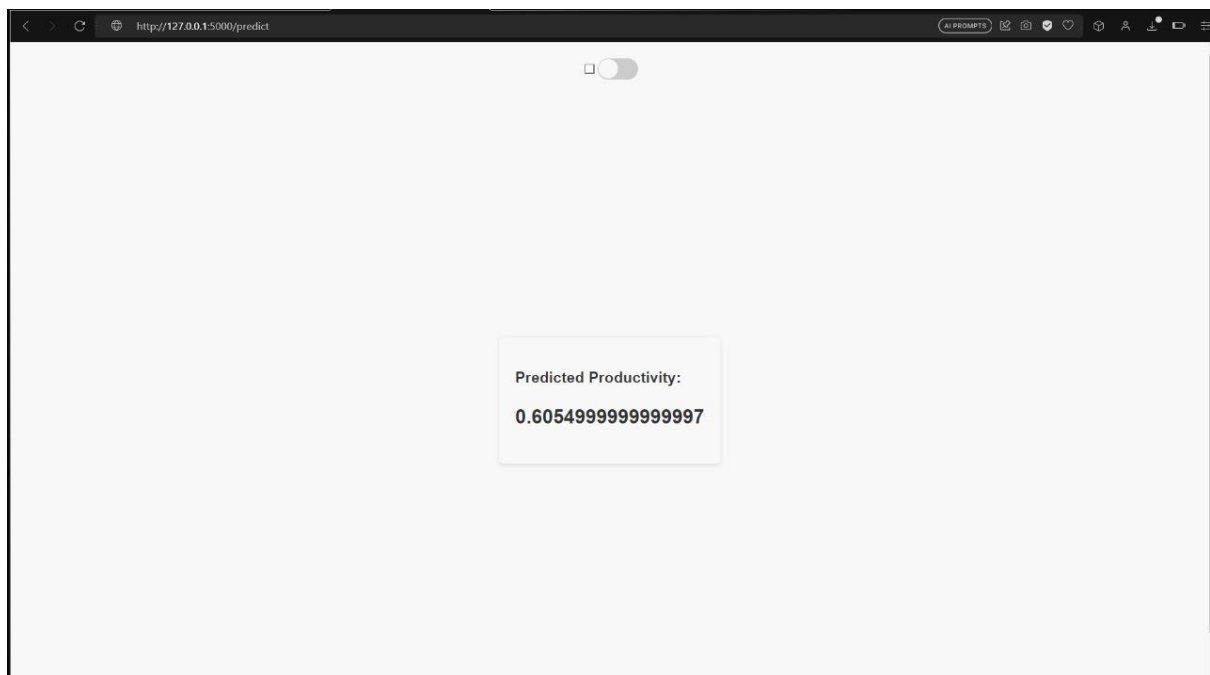
```

Implemented Output:



The screenshot shows a web browser window with the URL `http://127.0.0.1:5000/`. The page title is "Garment Productivity Prediction". The form contains the following elements:

- SMV:** A text input field containing the value "566".
- WIP:** A text input field containing the value "654".
- Overtime:** A text input field containing the value "678".
- Quarter:** A dropdown menu with "Q3" selected.
- Day:** A dropdown menu with "Monday" selected.
- Dark Mode:** A checkbox that is currently unchecked.
- Predict:** A dark button with the text "Predict".



Garment Productivity Prediction

SMV:
566

WIP:
654

Overtime:
678

Quarter:
Q3

Day:
Monday

☒ Dark Mode

Predict

