Data Collection and Preprocessing Phase

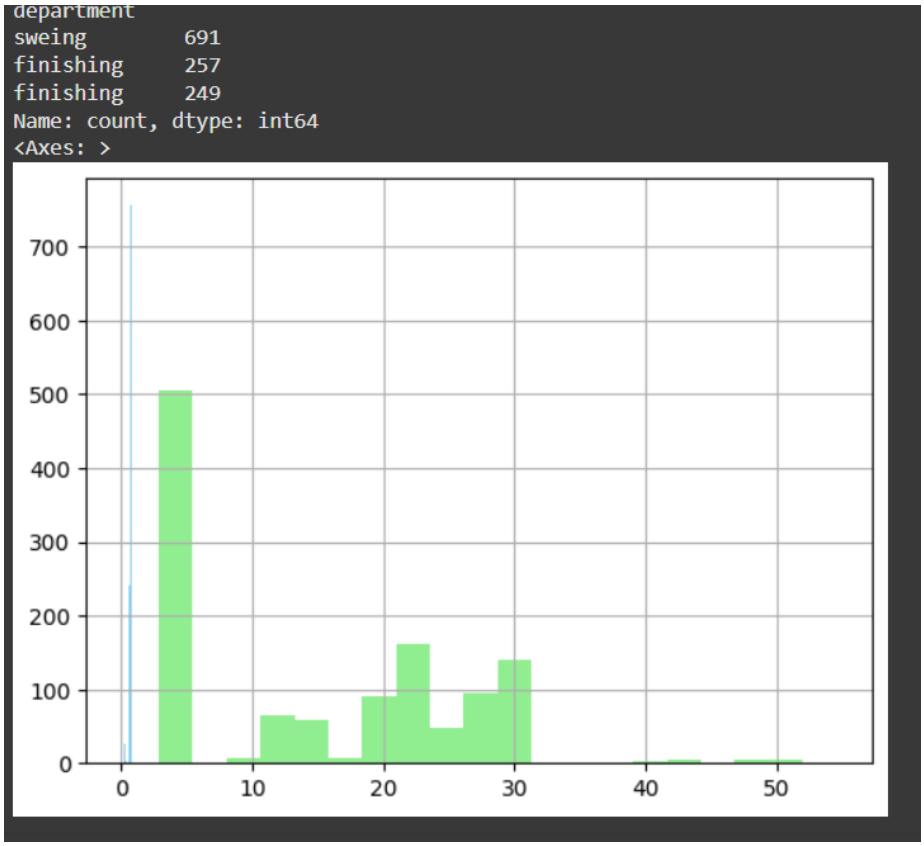
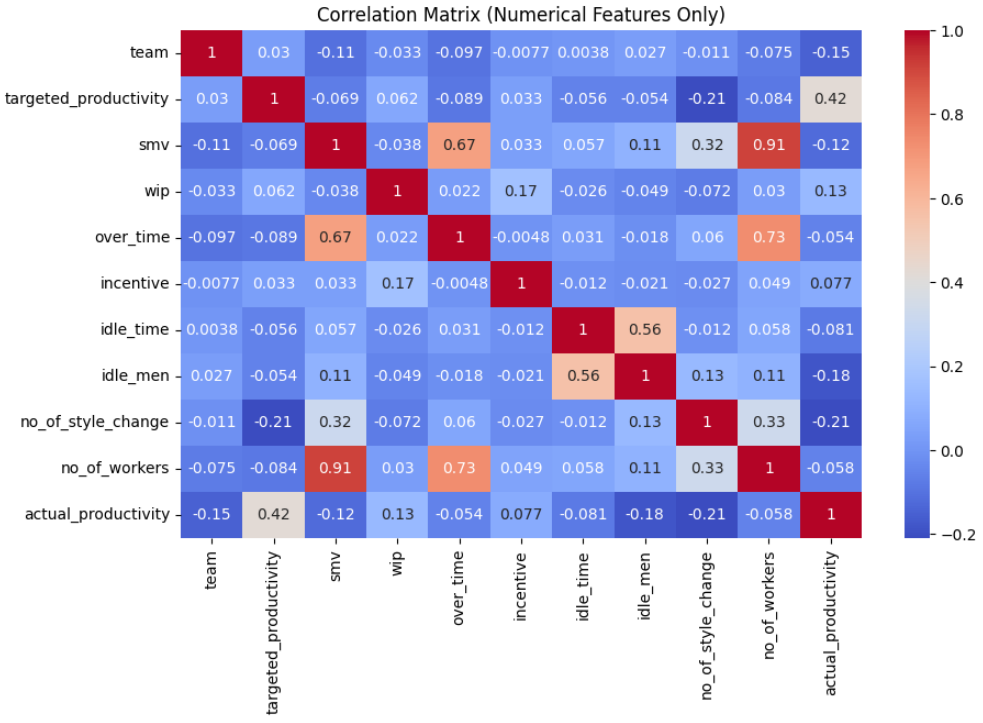
Date	24 June 2025
Team ID	SWUID20250177148
Project Title	Machine Learning Approach for Employee Performance Prediction
Maximum Marks	6 Marks

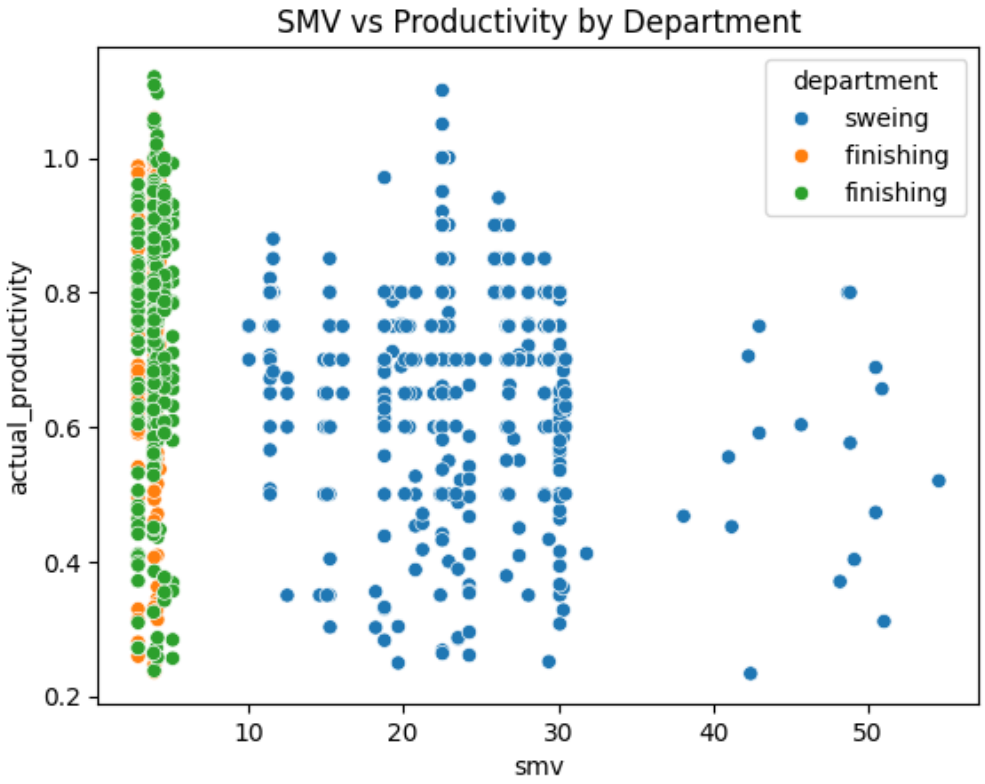
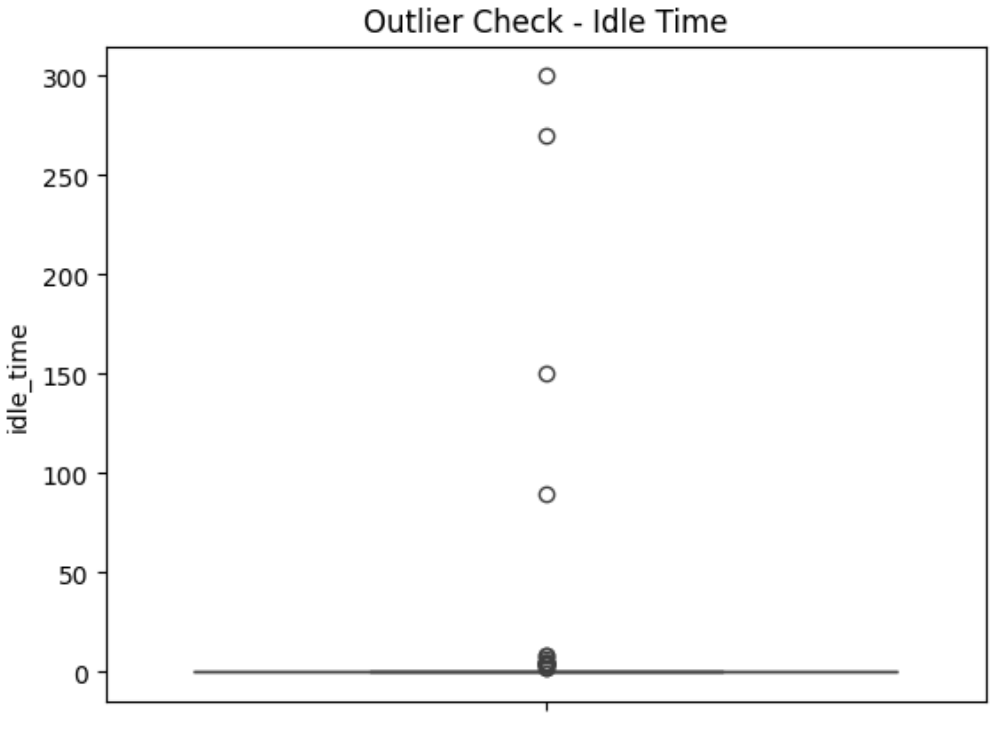
Data Exploration and Preprocessing

Dataset variables will be statistically analyzed to identify patterns and outliers.Python and Pandas were used for preprocessing tasks like encoding categorical variables, handling missing values, extracting date components, and normalizing inputs. This ensures model readiness with clean, structured, and numeric-only data.

Section	Description
Data Overview	**Dimension**: 1197 rows × 15 columns (including date and actual_productivity) **Descriptive statistics**

	team	targeted_productivity	smv	wip	over_time	incentive	idle_time	idle_men	no_of_style_change	no_of_workers	actual_productivity
count	1197.000000	1197.000000	1197.000000	691.000000	1197.000000	1197.000000	1197.000000	1197.000000	1197.000000	1197.000000	1197.000000
mean	6.426901	0.729632	15.062172	1190.465991	4567.460317	38.210526	0.730159	0.369256	0.150376	34.609858	0.735091
std	3.463963	0.097891	10.943219	1837.455001	3348.823563	160.182643	12.709757	3.268987	0.427848	22.197687	0.174488
min	1.000000	0.070000	2.900000	7.000000	0.000000	0.000000	0.000000	0.000000	0.000000	2.000000	0.233705
25%	3.000000	0.700000	3.940000	774.500000	1440.000000	0.000000	0.000000	0.000000	0.000000	9.000000	0.650307
50%	6.000000	0.750000	15.260000	1039.000000	3960.000000	0.000000	0.000000	0.000000	0.000000	34.000000	0.773333
75%	9.000000	0.800000	24.260000	1252.500000	6960.000000	50.000000	0.000000	0.000000	0.000000	57.000000	0.850253
max	12.000000	0.800000	54.560000	23122.000000	25920.000000	3600.000000	300.000000	45.000000	2.000000	89.000000	1.120437

Univariate Analysis	
Bivariate Analysis	

Multivariate Analysis	
Outliers and Anomalies	

Data Preprocessing Code Screenshots

Loading Data

```
[3] df = pd.read_csv('/content/garments_worker_productivity.csv')
    df.head()
```

	date	quarter	department	day	team	targeted_productivity	smv	wip	over_time	incentive	idle_time	idle_men	no_of_style_change	no_of_workers	actual_productivity
0	1/1/2015	Quarter1	sweing	Thursday	8	0.80	26.16	1108.0	7080	98	0.0	0	0	59.0	0.940725
1	1/1/2015	Quarter1	finishing	Thursday	1	0.75	3.94	NaN	960	0	0.0	0	0	8.0	0.886500
2	1/1/2015	Quarter1	sweing	Thursday	11	0.80	11.41	968.0	3660	50	0.0	0	0	30.5	0.800570
3	1/1/2015	Quarter1	sweing	Thursday	12	0.80	11.41	968.0	3660	50	0.0	0	0	30.5	0.800570
4	1/1/2015	Quarter1	sweing	Thursday	6	0.80	25.90	1170.0	1920	50	0.0	0	0	56.0	0.800382

Handling Missing Data

```
Shape of the dataset: (1197, 15)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1197 entries, 0 to 1196
Data columns (total 15 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   date                   1197 non-null   object
 1   quarter                1197 non-null   object
 2   department             1197 non-null   object
 3   day                    1197 non-null   object
 4   team                   1197 non-null   int64
 5   targeted_productivity  1197 non-null   float64
 6   smv                    1197 non-null   float64
 7   wip                    691 non-null    float64
 8   over_time              1197 non-null   int64
 9   incentive              1197 non-null   int64
 10  idle_time              1197 non-null   float64
 11  idle_men               1197 non-null   int64
 12  no_of_style_change     1197 non-null   int64
 13  no_of_workers          1197 non-null   float64
 14  actual_productivity    1197 non-null   float64
dtypes: float64(6), int64(5), object(4)
memory usage: 140.4+ KB

Missing values in each column:
date                       0
quarter                    0
department                 0
day                        0
team                       0
targeted_productivity      0
smv                        0
wip                      506
```

Data Transformation	Handling Date & Department
	`df['date'] = pd.to_datetime(df['date'])`
	`df['month'] = df['date'].dt.month`
	`df.drop(['date'], axis=1, inplace=True)`
	`df.head()  # ✅ OPTIONAL: Just to view updated data`

Feature Engineering	
Save Processed Data	Splitting data into train and test

	quarter	department	day	team	targeted_productivity	smv	wip	over_time	incentive	idle_time	idle_men	no_of_style_change	no_of_workers	actual_productivity	month
0	Quarter1	sweing	Thursday	8	0.80	26.16	1108.0	7080	98	0.0	0	0	59.0	0.940725	1
1	Quarter1	finishing	Thursday	1	0.75	3.94	NaN	960	0	0.0	0	0	8.0	0.886500	1
2	Quarter1	sweing	Thursday	11	0.80	11.41	968.0	3660	50	0.0	0	0	30.5	0.800570	1
3	Quarter1	sweing	Thursday	12	0.80	11.41	968.0	3660	50	0.0	0	0	30.5	0.800570	1
4	Quarter1	sweing	Thursday	6	0.80	25.90	1170.0	1920	50	0.0	0	0	56.0	0.800382	1

	quarter	department	day	team	targeted_productivity	smv	wip	over_time	incentive	idle_time	idle_men	no_of_style_change	no_of_workers	actual_productivity	month
0	0	2	3	8	0.80	26.16	1108.0	7080	98	0.0	0	0	59.0	0.940725	1
1	0	1	3	1	0.75	3.94	NaN	960	0	0.0	0	0	8.0	0.886500	1
2	0	2	3	11	0.80	11.41	968.0	3660	50	0.0	0	0	30.5	0.800570	1
3	0	2	3	12	0.80	11.41	968.0	3660	50	0.0	0	0	30.5	0.800570	1
4	0	2	3	6	0.80	25.90	1170.0	1920	50	0.0	0	0	56.0	0.800382	1

```
from sklearn.model_selection import train_test_split

# Features (X) = all columns except target
X = df.drop('actual_productivity', axis=1)

# Target (y) = what we want to predict
y = df['actual_productivity']

# Split into training (80%) and testing (20%) sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Check shapes
print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
```

```
X_train shape: (957, 14)
X_test shape: (240, 14)
```