WINTER 2024

BSMM 8750: PREDICTIVE MODELING & DECISION MAKING

CAPSTONE PROJECT REPORT

Strategic Optimization and Predictive Analysis for Enhanced Healthcare Operations - Greenshield

Submitted to:

Professor: Dr. Brent Furneaux

Submitted by:

Section 2

Group 1

Akhila Soodi Reddy (110117775)
Divyasree Geetha Balasubramanian (110117332)
Hoang Uyen Thu Le (110106222)
Vijaya Satwika Naidu Mandali (110116456)

Submission Date:

31st March 2024

Contents

1. Introduction

Green Shield Integrated Health Services, a subsidiary of Green Shield Canada, specializes in health and general claims processing, leveraging advanced technology to optimize claims management and enhance service delivery. With a focus on workforce analytics and operational diagnostics, Green Shield is committed to improving healthcare outcomes through innovative initiatives like the Central Medicine Program. Reporting significant annual revenues and serving a substantial Canadian demographic, the company is dedicated to employee engagement, efficient resource management, and reducing operational expenses, all while providing vital services like adjudicating claims and prior authorization for treatments, ensuring a high standard of care and service to its clients.

1.1 Abstract:

This report includes a thorough data analysis that was conducted for Greenshield, a business that is leading the way in utilizing analytics and technology to improve operations and service delivery in the healthcare industry. The primary goal of this project is to formulate practical suggestions that improve productivity, lower the expense efficiency ratio, precisely predict workforce needs, streamline claim processing times, and ultimately manage operational shrinkage. The process of analysis commenced with a thorough cleaning of the data to ensure the accuracy and consistency of the operational information being examined.

Through the strategic examination of historical operational data, this study employs advanced statistical methods and leverages Jupyter Notebook software tools to identify inefficiencies, model future scenarios, and recommend optimizations across various business units. In addition to diagnosing existing operational inefficiencies, this report provides a robust foundation for predictive modeling. The analysis aims to achieve a dual objective: significantly lower the expense efficiency ratio, thus reflecting more efficient use of resources, and simultaneously increase productivity without incurring additional operational costs.

1.2 Project Objectives:

The primary objective is to develop a data-driven understanding of how current resource allocation, scheduling practices, and task management contribute to overtime. This understanding will serve as the foundation for proposing targeted interventions designed to optimize workflows, improve resource distribution, and implement more effective scheduling techniques.

- Identify and analyse Key Performance Indicators (KPIs) affecting the performance of departments within Greenshield Integrated Healthcare Services.
- Analyse KPIs related to employee performance to assess and enhance individual productivity.
- Employ predictive analytics to optimize employee productivity without leading to burnout, ensuring efficient resource allocation, and staffing.
- Predict and analyse time taken for specific activities to identify inefficiencies and optimize processes for better resource allocation and scheduling.
- Use predictions to proactively schedule staff during high-activity periods, identify hiring needs, and address structural overloads that consistent overtime cannot mitigate.
- The goal is to establish a balanced operational model that not only reduces the need for overtime but also boosts overall productivity, ensuring that Greenshield Integrated Healthcare Services can deliver high-quality care more efficiently.

2. Data Preparation

The data preparation phase involved understanding the structure and quality of the data received, cleaning the data to ensure accuracy and completeness, and transforming it into a format suitable for analysis. This process is crucial for ensuring that the subsequent analysis is based on reliable and meaningful information.

2.1 Data Understanding:

We have received three excel files in 2 phases - Daily Summaries (Phase 1), Department Interval Production Results (Phase 2), Report for Departments Hours Summary (Phase 2). In this initial stage, we engaged in a thorough exploration of the datasets to ascertain the nature, scope, and quality of the data received. The datasets provided encompass detailed operational metrics from a variety of departments, covering daily summaries and departmental production hours. The diversity in metrics and departmental granularity helped us to get a comprehensive overview of organizational operations over time.

- **Daily Summaries**: This dataset offers a daily account of departmental turnovers, inventory levels, completed and received workloads, among other metrics, across multiple departments. This dataset is instrumental in tracking daily operations and assessing performance over time.
- **Department Interval Production Results:** This dataset provides detailed information about activities within departments, including shift start dates, interval identification, start times, employee IDs, activity types, and the duration of these activities in seconds. It's pivotal for understanding employee productivity and departmental efficiency on an interval basis.
- **Report for Departments Hours Summary**: This dataset aggregates hours spent on different types of activities by department. It categorizes hours into different types, providing a succinct summary of labour distribution and departmental focus.

2.2 Data Cleaning:

Data cleaning is a foundational process which entails a series of actions aimed at correcting, refining, and preparing raw data to ensure its quality and usability for analysis. In the "Report for Departments Hours Summary" during phase 2, upon conducting preliminary analysis steps, we encountered an anomalous scenario associated with Employee -1. This situation appears to pertain to an automated process rather than the activity of an individual. Consequently, to ensure an accurate assessment of employee performance, we have excluded this non-relevant data pertaining to Employee -1 from our analysis.

Identifying and Handling Missing Values:

Missing data can conceal the true analysis and result in biased conclusions. In our dataset, we detected missing values across various crucial columns. We addressed this issue by initially identifying the pattern and cause of the missing data. Where applicable, and following a thorough impact analysis, we filled in the missing values using statistical methods like imputation with mean, median, or mode.

- In phase 1:
    i. The Daily Summaries excel files contained more missing values (NaN). Where possible, missing values are imputed based on the mean, median, or mode of the respective column, or through more complex methods like interpolation. In cases where imputation may not be appropriate or where missing data represents a significant portion of the data for a specific variable, we removed the entries. This is particularly relevant if the missing data is not random but biased in some way.
    ii. We dropped the columns with a significant proportion of missing data (over 50%), such as Dept03 Turns Processing and Dept03 Date of Oldest, which both have 1686 missing values out of 2144 entries, we opted for dropping those columns. The rationale is that the high level of incompleteness would likely introduce bias or reduce the reliability of any imputation. For columns like dept 07 Turnaround Days and dept 07 Inventory methods like backward and forward fill are used to treat the missing values.
- In Phase 2, the received data has minimal missing values but requiring careful consideration of outliers and contextual relevance.

Correcting Data Anomalies:

- Used statistical methods or visualization tools to identify outliers. For numerical data, IQR (Interquartile Range) is used. Interquartile Range (IQR) method for numerical data, which involves calculating the range between the first (Q1) and third (Q3) quartiles. Data points falling below Q1 - 1.5 * IQR or above Q3 + 1.5 * IQR are considered outliers. In phase 1, we have used scatter plot to visualize the outliers for 'Dept10 Turnaround Days', this graphical tool aids in visual outlier detection by plotting 'Dept10 Turnaround Days' against the 'Oldest Date in Queue'.
- Once outliers are detected, a thorough investigation is conducted to determine whether they are errors or valid extreme values. Depending on the investigation's findings, outliers are treated using techniques such as imputation or capping. Imputation involves replacing outliers with suitable values based on the data distribution or domain knowledge, while capping involves setting thresholds to limit the impact of extreme values. In phase 2, the outliers found in the SumofHours column are treated with the help of relationships and patterns observed in relation to other columns in the dataset.

2.3 Data Transformation:

We have followed a few steps to make the data ready for analysis. In phase 1, we did Data transformation by standardizing date formats like for column 'Dept30 Oldest Date of Work', normalizing numerical data for comparability, encoding categorical variables for numerical analysis, and aggregating detailed data for simplicity. These transformations helped us in converting raw data into a format that facilitated comprehensive analysis and supported informed decision-making.

In phase 2, we try to standardize the data information such as 'DateofShiftStart' or 'Interval Start' into a consistent format, making it easier to manipulate, compare, and analyse time-based data. This normalization allowed us to compare performance across departments with different scales of operations.

3. Methodology

3.1. Exploratory data analysis:

In phase 2 of our analysis for the "Department Interval Productions" dataset, we identified the presence of Employee -1. Preliminary analysis suggests that Employee -1 might be associated with automated processes aimed at boosting operational efficiency, as illustrated in Figures 1 and 2. Analysis of the total transactions processed by employees reveals a distribution spanning from the morning to the early morning time block, with the morning and afternoon periods (6 AM to 6 PM) predominantly seeing a higher volume of transactions, as shown in Figure 3. Further examination highlights that the transactions conducted by employees likely necessitate specialized skills for transaction handling, indicated in Figure 4. Typically, the activity duration is set at 15 minutes per interval, representing the standard time allocation. However, we see examples of much shorter periods, such as less than 50 or 100 seconds, which might be due to micro-tasks such as logging or status checking as shown in (Fig 5).

For the 'ReportforDeptsHoursSummary' dataset, the histograms for "Distribution of Average Total Hours" and "Distribution of Average OT Hours" (Fig 7) provide a visualization of the workforce's time management. The average total hours are normally distributed, indicating a standardized workday for most employees. In contrast, the average overtime hours show a right-skewed distribution, where most employees log few overtime hours, while a small segment accrues higher overtime.

From Phase 2, the bar chart (Fig 8) illustrating the "Top 10 Performers" and "Bottom 10 Performers" in terms of efficiency percentage reveals a marked contrast in performance levels within the company. The top performers exhibit efficiency percentages that range significantly, with the highest performers operating close to full efficiency. Conversely, the bottom performers indicate a more compressed

range of lower efficiency, suggesting potential underutilization of skills or resources. This disparity signals opportunities for targeted training programs, redistribution of workload, or process refinement to bolster overall workforce productivity.

3.2. Trend Analysis:

For Greenshield, this technique was employed to dissect historical operational data and unearth underlying patterns that could signal future performance and opportunities for strategic decision-making. We applied moving averages to smooth out short-term fluctuations and highlight longer-term trends in key performance indicators, such as claim processing times and staff productivity levels. Our trend analysis focuses on overtime patterns, efficiency fluctuations, call volume trends within Greenshield's departments, and transaction trends by types, which provide invaluable foresight for strategic planning and resource allocation.

Transaction Trends Over Time for Types:

- Each type continues to show a distinct trend over time, which is now potentially more reflective of human employee activity. These trends may more accurately represent shifts in demand, production priorities, or operational strategies (Fig 6).

Overtime Trends by Department:

- The overtime analysis (Fig 9) within Greenshield has uncovered significant variations that demand a strategic organizational response. The analysis spans three years of operational data, presenting a clear narrative about workforce management and departmental efficiency. The graph (Fig 9) shows significant fluctuations in overtime hours across multiple departments over a period of several years.
- Notably, certain departments (e.g., Dept001 and Dept002) have exhibited peaks in overtime, these peaks are indicative of periods where the workload exceeded normal capacity, potentially due to project deadlines, seasonal demand increases, or other operational pressures. A pattern worth noting is the regular occurrence of end-of-quarter or end-of-year peaks in overtime across several departments.

Comparison of Call Volumes Between Departments (Phase 1):

- In (Fig 11) an overlay of total call volumes for two departments from phase 1 emphasizes contrasting trends. Dept05 consistently handles a higher volume of calls compared to Dept11, suggesting a heavier workload or possibly more efficient call management. This comparison can be instrumental in resource planning and in implementing best practices from higher volume departments to those with fewer calls.

Work Efficiency Variations by Shift Hours:

Start time:

- Efficiency trends, broken down by start hours (Fig 10) indicates higher efficiency at specific points in the early hours. The graph showed distinct peaks in efficiency at certain start hours, suggesting that shifts beginning at these hours have higher average efficiency percentages.

End time:

- In contrast, efficiency appears to decline towards the end of shifts (Fig 10), particularly towards the later hours of the day. This insight could lead to recommendations for shift adjustments or break allocations to sustain high efficiency levels throughout the workday. These graphs suggest that Greenshield might benefit from optimizing shift schedules.

3.3. Tools used for statistical Analysis:

For statistical analysis, we employed a dual-tool approach, leveraging both Python and Microsoft Excel to harness their respective strengths in handling complex statistical tasks and initial data manipulations. Python, with its vast libraries such as Pandas for data manipulation, NumPy for numerical computation, and Matplotlib and Seaborn for visualization, served as the backbone for in-depth exploratory data analysis, statistical testing, and model building.

Microsoft Excel complemented this by providing a user-friendly platform for preliminary data cleaning, quick exploratory analysis through pivot tables and charts, and utilizing formula functions for basic computations and data transformations. This synergistic use of Python's powerful data analysis capabilities and Excel's intuitive interface and versatility allowed for a comprehensive and efficient analysis process, ensuring a thorough investigation, and understanding of the dataset at hand.

3.4. Correlation analysis:

Correlation between Scheduled Breaks and Work Efficiency:

- The correlation analysis (Fig 12) was the correlation between the number of scheduled breaks and work efficiency. The corresponding scatter plot demonstrated a strong negative correlation (r = -0.86), with a clear downward trend indicating that as the number of scheduled breaks increases, there is a substantial decrease in efficiency percentage.
- Perhaps a few breaks improve efficiency, but beyond a certain point (1.00 hour), additional breaks have the opposite effect. This strong correlation could imply that more frequent breaks disrupt workflow and negatively impact overall efficiency. However, it is important to note that correlation does not imply causation, and further analysis is required to determine the underlying factors contributing to this relationship.

Correlation between Start Time and Work Efficiency:

- The scatter plot (Fig 13) presents a dispersed arrangement of data points with a correlation coefficient (r) of -0.02, suggesting a negligible negative linear relationship. This implies that the start time of shifts has almost no linear correlation with the efficiency levels of employees.

Correlation between End Time and Work Efficiency:

- The correlation coefficient (Fig 13) is slightly more negative at -0.16, indicating a weak negative linear relationship. There appears to be a slight tendency for later end times to correlate with lower efficiency percentages, possibly due to longer workdays leading to fatigue. This correlation analysis of start and end time to work efficiency can help the company to schedule the shifts which gives the maximum work efficiency.

Correlation Matrix:

- The correlation matrix (Fig 14) for Department 04 operational metrics highlights key relationships. Notably, a very strong positive correlation exists between 'Dept04 Total Inventory BreakOut2' and 'Dept04 Total Inventory BreakOut3' (r = 0.99), suggesting a significant link between these inventory metrics. 'Dept04 Ready Inventory' is closely associated with 'Dept04 Total Inventory' (r = 0.81), underscoring a direct relationship between available inventory and overall stock levels.
- Interestingly, 'Dept04 Hours' show minimal impact on productivity rates, with a near-zero correlation (r = -0.04), indicating that longer hours do not necessarily equate to higher productivity. The slight negative correlation between 'Dept04 Productivity Rate' and 'Dept04 Errors' (r = -0.18) implies that more errors may slightly affect productivity. These insights inform areas that may benefit from targeted strategies to enhance efficiency and accuracy in departmental operations.

4. Model Building

In our quest to predict and manage overtime trends within Greenshield, we constructed a Seasonal Autoregressive Integrated Moving Average with eXogenous variables (SARIMAX) model. This choice was informed by the model's ability to account for both the non-stationary aspects of our time series data and the seasonality observed in historical overtime patterns. The model was meticulously calibrated using historical overtime data, factoring in seasonal fluctuations to accurately capture the underlying trends and repetitive cycles over time.

We forecasted overtime work hours for all the departments together (Fig 17) and, we forecast the overtime hours for Dept 006 (Fig 15) as it has a high average hours of overtime per employee compared to other department employees (Fig 16). So, this forecast reveals upcoming high-workload periods for Dept 006, necessitating advance preparation for uninterrupted operations.

Additionally, we forecast the total transactions using SARIMAX model (Fig 18). The historical data shows variability in transactions with noticeable peaks and troughs. This could indicate weekly cycles or other recurring patterns within the transaction data. The model was trained on the historical transaction data, with particular attention paid to capturing the recurrent spikes in transaction volumes. In crafting our model, we placed a particular emphasis on not only capturing the inherent trends and seasonality present in the historical data but also ensuring the model's adaptability to potential anomalies and irregularities.

4.1 Model Selection:

For Greenshield’s transaction and overtime data, which exhibits clear periodicity and trend behaviour, SARIMAX's ability to incorporate seasonality and trend into its forecasting made it the optimal choice. The SARIMAX model's parameters were chosen based on the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC), which provided a quantitative measure for model comparison, considering the trade-off between model complexity and fit. After experimenting with various parameter combinations, the model with the lowest AIC and BIC values was selected, ensuring the best fit while avoiding overfitting.

The fluctuations in overtime indicate not only a dependency on historical data but also suggest periodicity, which SARIMAX models are particularly adept at capturing. This is evident in the ability of the forecasted data (red line) to mirror the historical trends (blue line), signifying that SARIMAX can effectively factor in both the seasonality and the non-stationary nature of overtime occurrences. Such predictive accuracy is vital for Greenshield's operational planning, allowing for anticipatory adjustments in workforce management to address predicted increases in overtime demand.

The forecast of total transactions is done using SARIMAX model (Fig 18). This decision was driven by the model’s ability to handle the volatility and periodicity inherent in the transaction records.

4.2 Model Evaluation:

In evaluating the SARIMAX model used to forecast overtime at Greenshield, we employed a combination of statistical accuracy measures and practical applicability tests. The SARIMAX model underwent a rigorous evaluation process to ensure its predictive accuracy and reliability. The model's forecasts were subjected to time series cross-validation, with a keen focus on error metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE), all of which returned values indicating high accuracy and a tight fit to historical data. We got the RSME value of 1183 for SARIMAX model to forecast total transactions (Fig 18) and 668 for total overtime hours prediction.

The residuals analysis affirmed the model's appropriateness, showing no significant patterns and confirming that the seasonal and trend components of the overtime data were well-captured. The

model outperformed simpler benchmarks and aligned closely with Greenshield's operational cycles, underscoring its value in effective workforce planning and strategic decision-making.

5. Results and Findings

Our analysis reveals intriguing patterns related to shift timings, departmental work efficiencies, break schedules, and overtime hours that have important implications for operational management.

Shift Timings and Efficiency:

- The investigation into shift starts and end times (Fig 10) across various departments highlights critical time slots that align with peak efficiency. Data indicates that shifts commencing at around 6:30 AM, as well as in the evening at 8 PM and 11 PM, correspond with higher average efficiency percentages, hinting at a potential correlation between these start times and heightened productivity.
- Moreover, the analysis of end times revealed a noteworthy peak in efficiency for shifts concluding around 3 PM (Fig 10). These findings suggest that mid-afternoon end times might be optimal before efficiency noticeably declines, particularly for shifts ending late in the evening.

Departmental Work Efficiency:

- We have found the work efficiency metric of each department by using the formula:

$$Work\ efficiency\ = \frac{\text{Pr}\,o\ duction\ hours}{(Regular\ hours + Overtime\ hours)}$$

- A comparative review of departmental efficiencies (Fig 19) over the same period presents a homogeneous landscape with minor variations, indicating that average efficiency levels are relatively consistent across departments. This suggests that, from a productivity standpoint, departments are performing comparably. But this metric helped us with our main goal of how we can improve work efficiency.

Impact of Scheduled Breaks on Efficiency:

- The findings from our correlation analysis (Fig 12) reveal a compelling dynamic between scheduled breaks and workplace efficiency, shedding light on the intricate balance required to optimize productivity.
- Our correlation analysis focusing on scheduled breaks and efficiency uncovers a strong negative relationship, with an R-value of -0.86. The trend is unmistakably downward, suggesting a significant inverse relationship between the number of scheduled breaks and efficiency percentages. Interestingly, efficiency seems to be optimized with fewer breaks, with a threshold (1:00 hour) beyond which additional breaks appear to be counterproductive. This threshold suggests that the optimal scheduling and duration of breaks are crucial for maximizing efficiency.

Overtime hours Analysis:

- Lastly, our overtime analysis across departments over a three-year span exhibits stark discrepancies, particularly with Department 006 (Fig 16), which consistently reports substantially higher average overtime hours per employee for three continuous years. This indicates potential inefficiencies or resource allocation issues that require immediate attention.
- The sustained high overtime could suggest that Department 006 (Fig 16) has systemic issues with workload balancing, where the volume of work regularly surpasses the capacity of the current staffing levels.

- Conversely, department 001 consistently showed lower average overtime hours per employee (Fig 16) for the consecutive 3 years, setting a standard of operational efficiency that could serve as a benchmark for other departments.

6. Recommendations:

After an exhaustive analysis of Greenshield's operational data, we put forward a series of strategic recommendations intended to bolster operational efficiency, refine resource deployment, and judiciously manage operational expenditures.

Optimization of Shift Scheduling:

- Adjust shift commencement times to correspond with the identified peak productivity windows at 7 AM, 8 PM, and 11 PM. Leveraging these times can maximize employee productivity by syncing work hours with natural high-performance intervals.
- Standardize work shifts to conclude by 3 PM to exploit the elevated efficiency rates sustained during the first half of the operational day. Analyse the practicality and impact of curtailing later shifts to deter a potential dip in productivity.

The optimal shift models identified as:

- **Morning Shift:** 7 AM to 3 PM, capturing the fresh energy of early hours for an 8-hour efficient work window.
- **Late Afternoon Shift:** 4 PM to 12 AM, targeting the second wind of productivity commonly experienced later in the day.
- **Night Shift:** 8 PM to 4 AM, accommodating nocturnal operations, crucial for continuous service delivery.

Proactive Overtime Management:

- Delve into the causative factors behind the escalated overtime in Department 006. Scrutinize the possibility of redistributing tasks, augmenting workforce capacity, or remodelling workflow processes to mitigate the dependency on overtime work.
- Integrate predictive analytics, exemplified by the application of the SARIMAX model, to anticipate high-activity intervals and pre-emptively manage staffing. This approach can diminish the incidence of unscheduled overtime, fostering a more balanced work environment and potentially yielding cost savings.

Benchmarking and Best Practices:

- Utilize the data from Department 001 as a benchmark to identify best practices in managing overtime and efficiently distributing workload.
- Promote knowledge sharing across departments to create a unified approach towards operational efficiency, ensuring that successful strategies are communicated and adopted organization wide.

Workforce Development and Support:

- Enhance workforce training programs to ensure all employees are adept at managing varied workloads and can maintain productivity under different shift models.
- Invest in employee support systems that provide resources for time management, stress management, and wellness, recognizing that a supported employee is a productive employee.

Break Scheduling Strategy:

- Undertake a comprehensive review of the current break schedule to establish an equilibrium that fosters high productivity while allowing adequate rest. Enforce a structured break regime that avoids surpassing the threshold 1 hour mark as indicated by our analysis, to preclude efficiency downturns.
- Shorter, more frequent breaks are often lauded for their ability to refresh and refocus the mind, reducing cognitive load and mitigating fatigue. Conversely, longer breaks, especially those exceeding the identified threshold, might disrupt workflow, leading to a slower restart and a potential decrease in momentum and focus.
- Initiate a pilot program with revised break schedules, subsequently assessing the resultant effects on productivity levels and workforce morale, fine-tuning the approach based on feedback and data analytics.

Continuous Monitoring and Improvement:

- Establish a monitoring framework to regularly assess the impact of the new shift schedules and break policies on operational metrics.
- Develop a responsive feedback loop that allows for quick adaptation and refinement of practices based on real-time operational data, employee feedback, and evolving business needs.

By implementing these measures, Greenshield can streamline its operations, thereby driving enhanced productivity, fostering a more satisfied workforce, and ultimately guaranteeing an enhanced level of service to its clients.

7. Conclusion

In conclusion, the extensive data analysis conducted for Greenshield Integrated Health Services has provided valuable insights into the company's operational dynamics, employee productivity, and resource management. Our findings highlight key areas where strategic interventions can significantly enhance efficiency and reduce reliance on overtime, contributing to a more balanced and cost-effective operational model. Our findings reveal that strategic adjustments to shift start and end times can significantly enhance efficiency. The data suggest that certain times of the day are more conducive to high productivity, and aligning work schedules to these periods can yield substantial benefits.

Departmental comparisons have provided a valuable benchmarking opportunity, showcasing Department 001 as an exemplar of operational efficiency. On the other hand, the stark contrast in overtime hours, especially within Department 006, calls for a deeper examination of operational processes and resource distribution. The predictive modeling undertaken, specifically using the SARIMAX model, stands out as a beacon for proactive workforce management. It demonstrates the potential to forecast future needs accurately, allowing Greenshield to prepare for high-demand periods without excessive reliance on overtime.

To conclude, continuous monitoring, adaptability, and a commitment to data-centric optimization will be crucial in sustaining and building upon the gains achieved. The foundation laid by this analysis is robust, this report underscores the importance of data-driven decision-making in operational management. The proposed recommendations, derived from a thorough analysis, have the potential to create significant improvements in operational efficiency. By optimizing shift schedules to align with peak productivity periods, revising break strategies to bolster efficiency, and proactively managing overtime through predictive analytics, Greenshield is poised to enhance its operational efficacy significantly.

8. References:

- Python Software Foundation. (2020). Python Language Reference, version 3.7. Available at https://www.python.org

- McKinney, W. (2010). Data Structures for Statistical Computing in Python. In S. van der Walt & J. Millman (Eds.), Proceedings of the 9th Python in Science Conference (pp. 51-56).
- Waskom, M. L. (2021). Seaborn: statistical data visualization. Journal of Open Source Software, 6(60), 3021. https://doi.org/10.21105/joss.03021
- The pandas development team. (2020). pandas-dev/pandas: Pandas. Zenodo. https://doi.org/10.5281/zenodo.3509134
- Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *Time series analysis: Forecasting and control* (5th ed.). John Wiley & Sons.

9. Appendix

Fig 1: Top Employees by Transaction

```
from datetime import datetime
import numpy as np
import pandas as pd
import seaborn as sns

#1. Load the dataset
data =  pd.read_excel('/file_path/DeptartmentIntervalProductionResults.xlsx')
# 2. Correlation between employees (EmployeeID) and number of transactions
employee_transaction_corr =
data.groupby('EmployeeID')['Transactions'].sum().sort_values(ascending=False).reset_index()

#3. Load the chart
plt.figure(figsize=[14, 6])
top_employees = employee_transaction_corr.head(10)  # Top 10 for clarity
sns.barplot(x='EmployeeID', y='Transactions', data=top_employees, palette='coolwarm')
plt.title('Top 10 Employees by Total Transactions')
plt.xlabel('Employee ID')
plt.ylabel('Total Transactions')
plt.show()
```

Fig 2: Transaction by Time block for Top Employee

```
# Convert 'DateofShiftStart' to datetime format
data['DateofShiftStart'] = pd.to_datetime(data['DateofShiftStart'])

# Extract hour from 'Interval Start' for correlation analysis with 'Transactions'
# Convert 'Interval Start' to datetime to extract the hour part
data['Hour'] = pd.to_datetime(data['Interval Start'], format='%I:%M:%S %p').dt.hour
# Detailed Employee Analysis: Focus on the top performing employees and EmployeeID -1
top_employee_ids = employee_transaction_corr.head(10)['EmployeeID']  # Top 10 employees
employee_detail_analysis = data[data['EmployeeID'].isin(top_employee_ids)]

# Group by EmployeeID and Time Block for pattern analysis
employee_time_transactions = employee_detail_analysis.groupby(['EmployeeID', 'Time
Block'])['Transactions'].sum().unstack().fillna(0)

# Visualization: Employee Transactions by Time Block
plt.figure(figsize=[14, 8])
sns.heatmap(employee_time_transactions, annot=True, fmt=".0f", cmap="YlGnBu", linewidths=.5)
plt.title('Transactions by Time Block for Top Employees')
plt.xlabel('Time Block')
plt.ylabel('Employee ID')
plt.show()
```

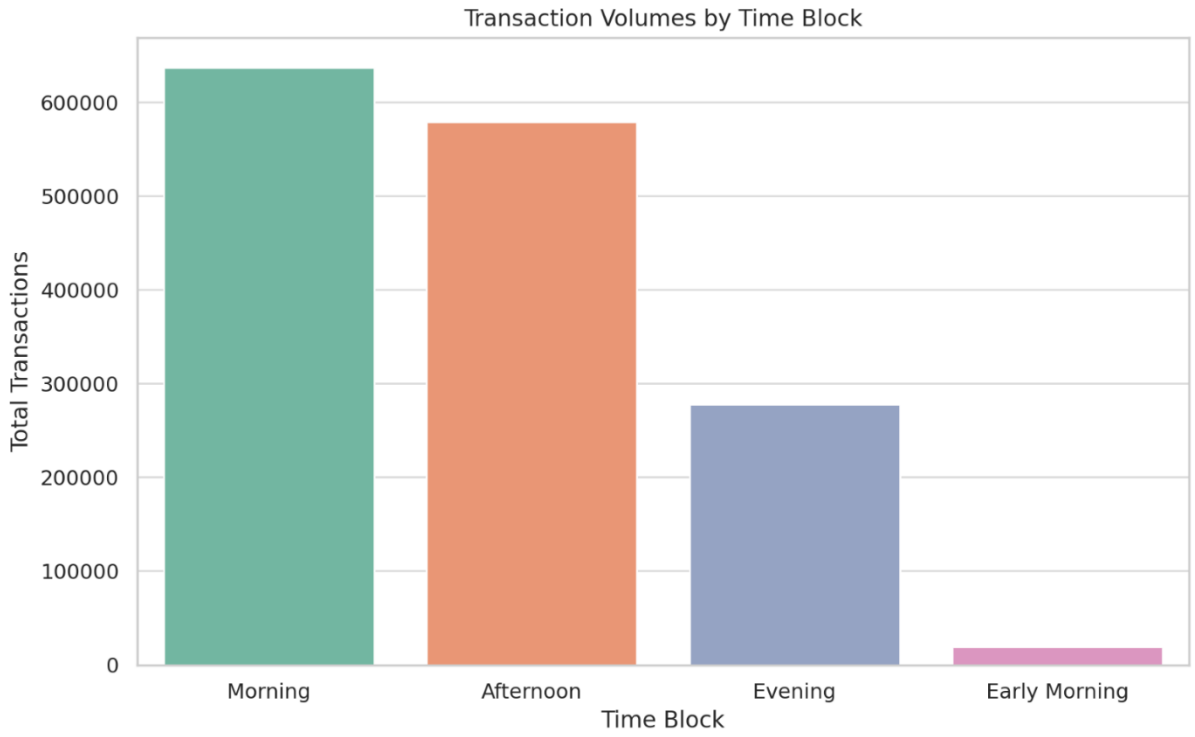

Fig 3: Transaction Volumes by Time Block

```
# Define time blocks for the analysis
time_blocks = {
    'Early Morning': (0, 6),
    'Morning': (6, 12),
    'Afternoon': (12, 18),
    'Evening': (18, 24)
}

# Assign each hour to a time block
def assign_time_block(hour):
    for block, (start, end) in time_blocks.items():
        if start <= hour < end:
            return block
    return 'Unknown'

data['Time Block'] = data['Hour'].apply(assign_time_block)

# Analyze transaction volumes within these time blocks
time_block_transactions = data.groupby('Time
Block')['Transactions'].sum().reset_index().sort_values(by='Transactions', ascending=False)

# Visualization: Transaction Volumes by Time Block
plt.figure(figsize=[10, 6])
sns.barplot(x='Time Block', y='Transactions', data=time_block_transactions, palette='Set2')
plt.title('Transaction Volumes by Time Block')
plt.xlabel('Time Block')
plt.ylabel('Total Transactions')
plt.show()
```

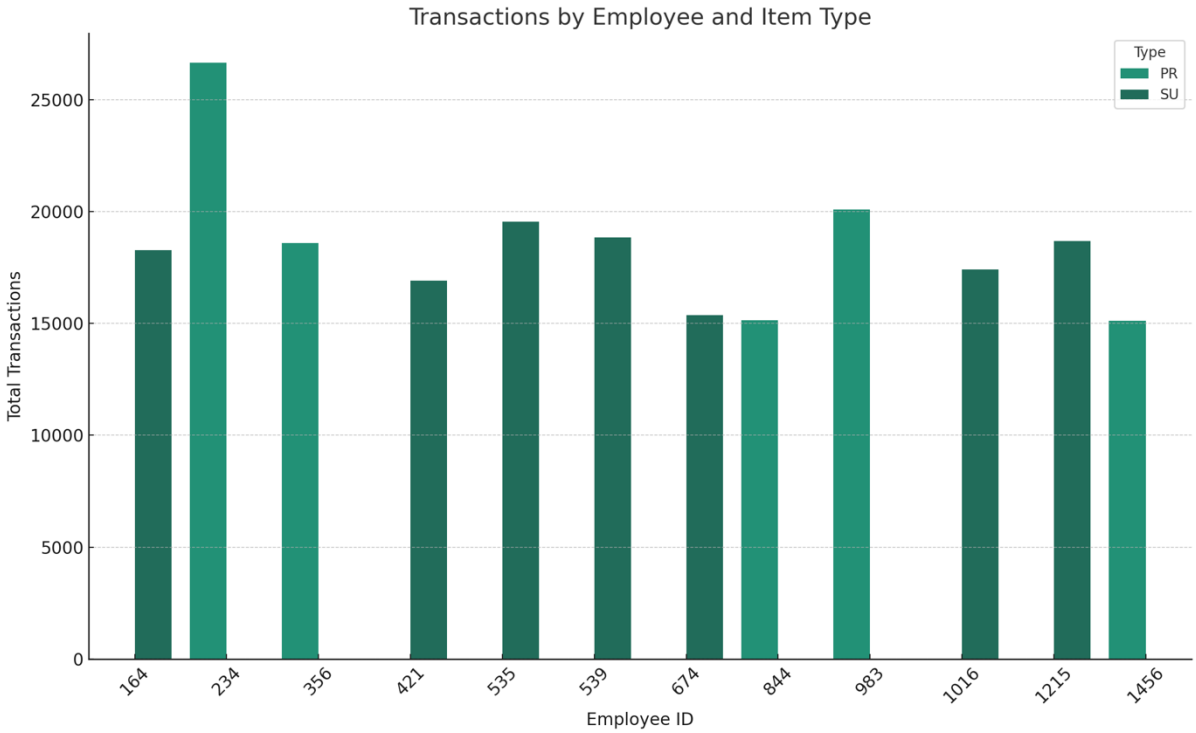

Fig 4: Transaction by Employee and Item Type

```
#Filter out EmployeeID -1

data_filtered = data[data['EmployeeID'] != -1]

#Total transaction per Employee and Type

top_items_per_employee = data_filtered.groupby(['EmployeeID',
'Type])['Transactions'].sum().reset_index()

# Finding the top item for each employee based on transactions

top_items_per_employee = top_items_per_employee.sort_values(['EmployeeID', 'Transactions'],
ascending=[True, False])

top_items_per_employee = top_items_per_employee.drop_duplicates(subset=['EmployeeID'],
keep='first')

# Visualizing the top item for the first 10 employees as a sample

top_items_sample = top_items_per_employee.head(10)

plt.figure(figsize=(14, 8))

sns.barplot(x='EmployeeID', y='Transactions', hue='Item Desciption', data=top_items_sample,
dodge=False, palette="viridis")

plt.title('Transaction by Employee and Item Type')

plt.xlabel('Employee ID')

plt.ylabel('Total Transactions')

plt.legend(title='Type', bbox_to_anchor=(1.05, 1), loc='upper left')
```

plt.xticks(rotation=45)

plt.show()

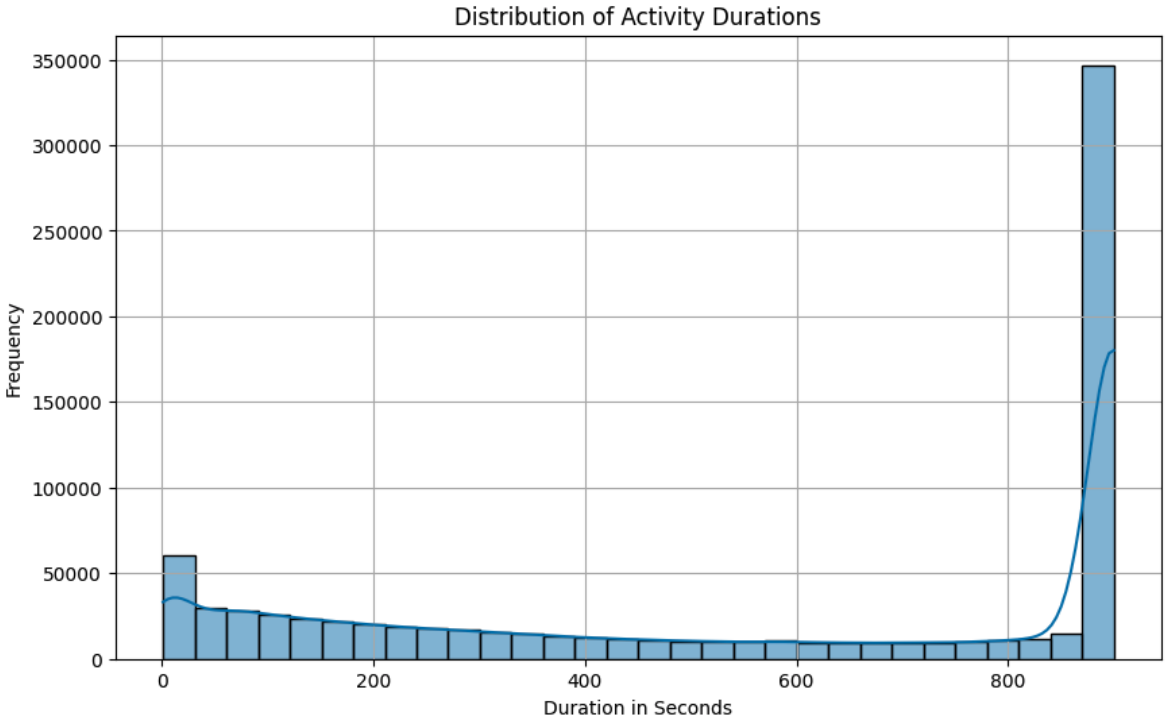

Fig 5: Activity Durations

#Load dataset

data_activity = pd.read_excel('/file_path/ IntervalEmpoyeeTimeTracking)

 # Filter out rows with missing 'Activity Type' or 'ActivitySeconds' to ensure quality analysis

data_filtered = data_activity.dropna(subset=['Activity Type', 'ActivitySeconds'])

 # Basic statistics for activity durations across all activities

overall_stats = data_filtered['ActivitySeconds'].describe()

 # Calculate statistics for each activity type

activity_stats = data_filtered.groupby('Activity Type')['ActivitySeconds'].describe()

 # Plotting the distribution of activity durations

plt.figure(figsize=(10, 6))

sns.histplot(data_filtered['ActivitySeconds'], bins=30, kde=True)

plt.title('Distribution of Activity Durations')

plt.xlabel('Duration in Seconds')

plt.ylabel('Frequency')

plt.grid(True)

plt.show()

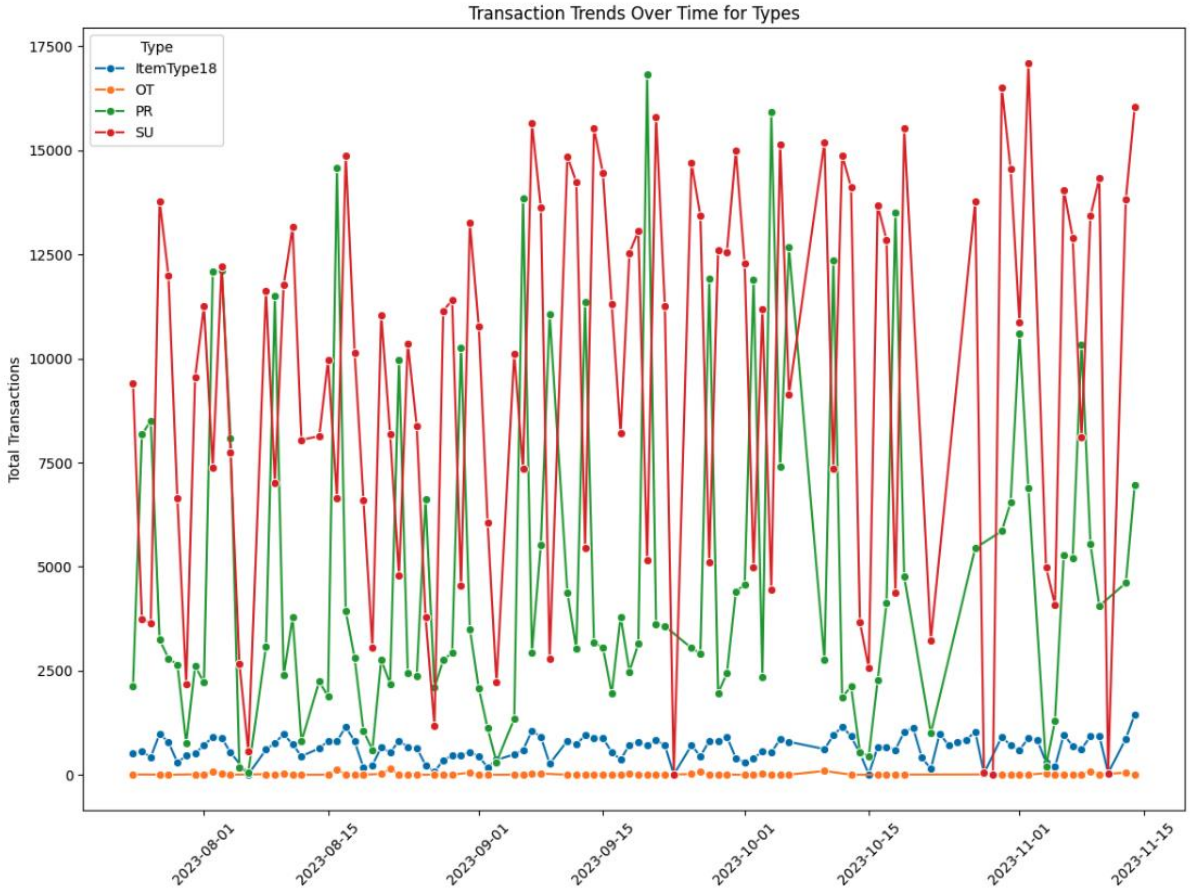

Fig 6: Transaction Trend for Types

 # Aggregate transactions by Date and Type, excluding transactions by Employee -1

type_date_transactions_excluding_minus1 = data_filtered.groupby(['DateofShiftStart', 'Type'])['Transactions'].sum().reset_index()

 # Continue to focus on the top types based on overall transaction volume, excluding Employee -1 transactions

top_types_excluding_minus1 = data_filtered.groupby('Type')['Transactions'].sum().nlargest(5).index

 # Filter the dataset to include only transactions for these top types, excluding Employee -1 transactions

top_type_transactions_over_time_excluding_minus1 = type_date_transactions_excluding_minus1[type_date_transactions_excluding_minus1['Type'].isin(top_types_excluding_minus1)]

 # Visualize the trend of transactions for these top types over time, excluding Employee -1 transactions

plt.figure(figsize=(14, 10))

sns.lineplot(x='DateofShiftStart', y='Transactions', hue='Type', data=top_type_transactions_over_time_excluding_minus1, marker='o')

plt.title('Transaction Trends Over Time for Types')

plt.xlabel('Date of Shift Start')

plt.ylabel('Total Transactions')

plt.xticks(rotation=45)

plt.legend(title='Type')

plt.show()

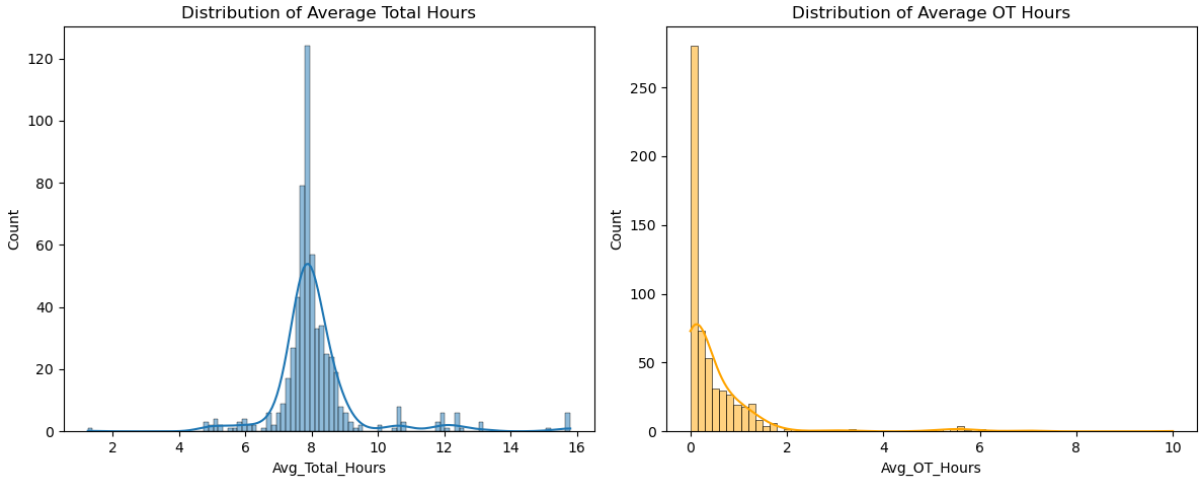

Fig 7: Comparative Distribution of Average Total Hours and Average Overtime Hours

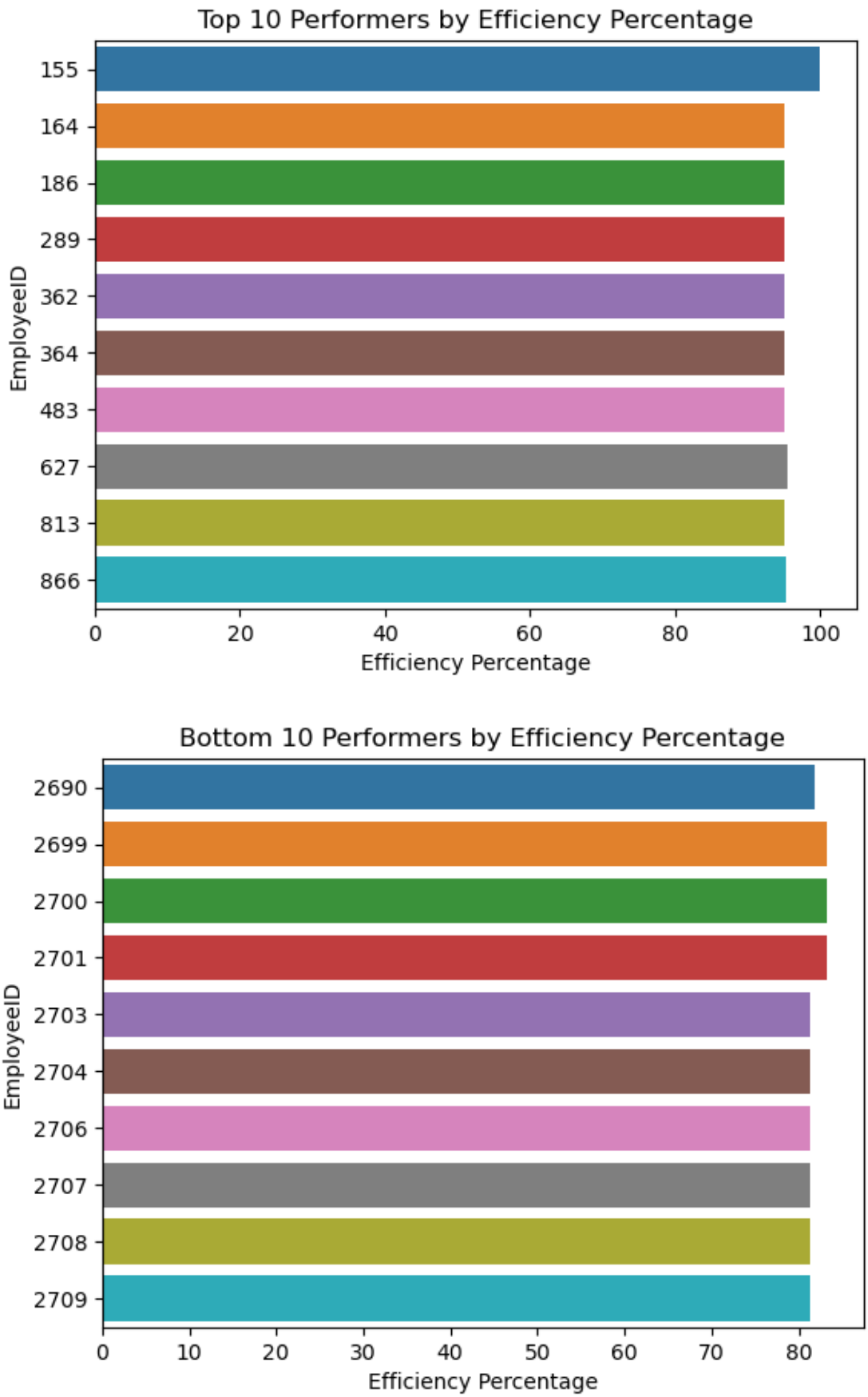

Fig 8. Top 10 Performers by Efficiency Percentage and Bottom 10 Performers by Efficiency Percentage

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load your data
data = pd.read_excel('C:/Users/Sreeja/OneDrive/Desktop/PMD/Production.xlsx')

# Calculate the average efficiency for each employee
employee_efficiency = data.groupby('EmployeeID')['Efficiency Percentage'].mean().reset_index()

# Sort to find top and bottom performers
top_performers = employee_efficiency.sort_values(by='Efficiency Percentage',
ascending=False).head(10)
bottom_performers = employee_efficiency.sort_values(by='Efficiency Percentage').head(10)

# Plotting top performers
sns.barplot(x='Efficiency Percentage', y='EmployeeID', data=top_performers, orient='h')
plt.title('Top 10 Performers by Efficiency Percentage')
plt.show()

# Plotting bottom performers
sns.barplot(x='Efficiency Percentage', y='EmployeeID', data=bottom_performers, orient='h')
plt.title('Bottom 10 Performers by Efficiency Percentage')
plt.show()
```

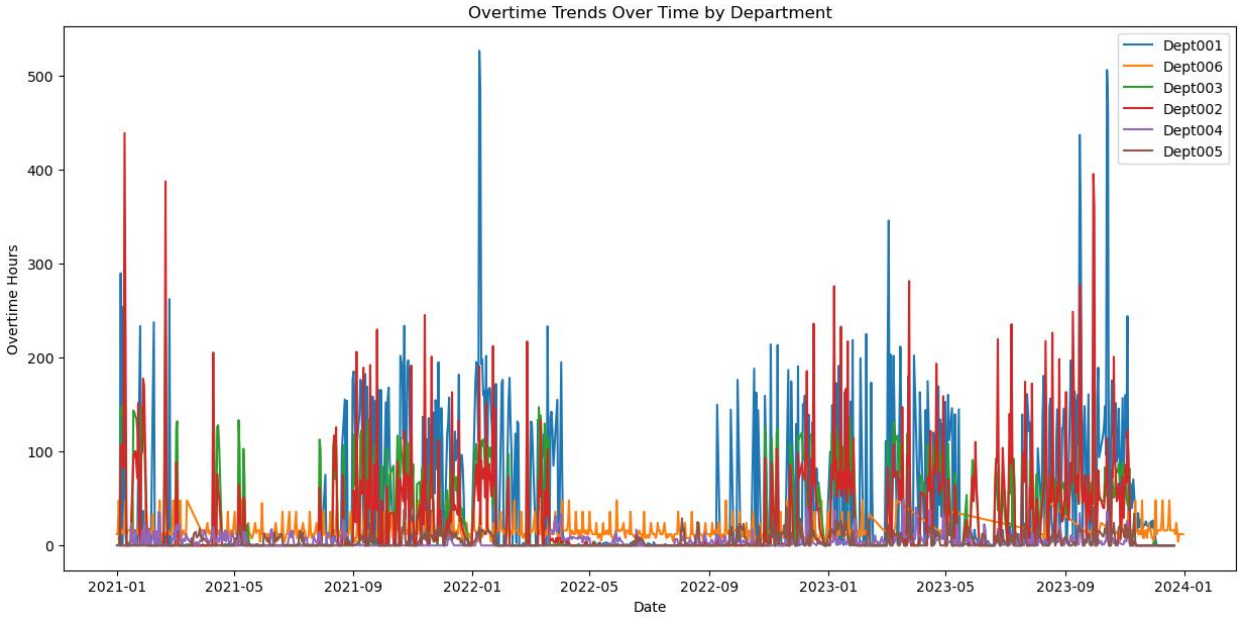

Fig 9: Departmental Overtime Hours Trend Analysis

```
import matplotlib.pyplot as plt

import seaborn as sns

import pandas as pd

# Calculate the average efficiency percentage for each department

department_efficiency = data.groupby('Dept ID')['Efficiency Percentage'].mean().reset_index()
```

```
# Sort the departments by their average efficiency for a meaningful visualization

department_efficiency_sorted = department_efficiency.sort_values('Efficiency Percentage', ascending=False)

# Visualization

plt.figure(figsize=(12, 8))

bar_plot = sns.barplot(x='Efficiency Percentage', y='Dept ID', data=department_efficiency_sorted, palette='viridis')

plt.title('Departmental Efficiency Comparison')

plt.xlabel('Average Efficiency Percentage')

plt.ylabel('Department ID')

# Adding annotations to each bar

for p in bar_plot.patches:

    width = p.get_width()

    plt.text(5 + width, p.get_y() + p.get_height() / 2,

             '{:1.2f}%'.format(width),

             ha='center', va='center')

plt.tight_layout()

# Display the plot

plt.show()
```

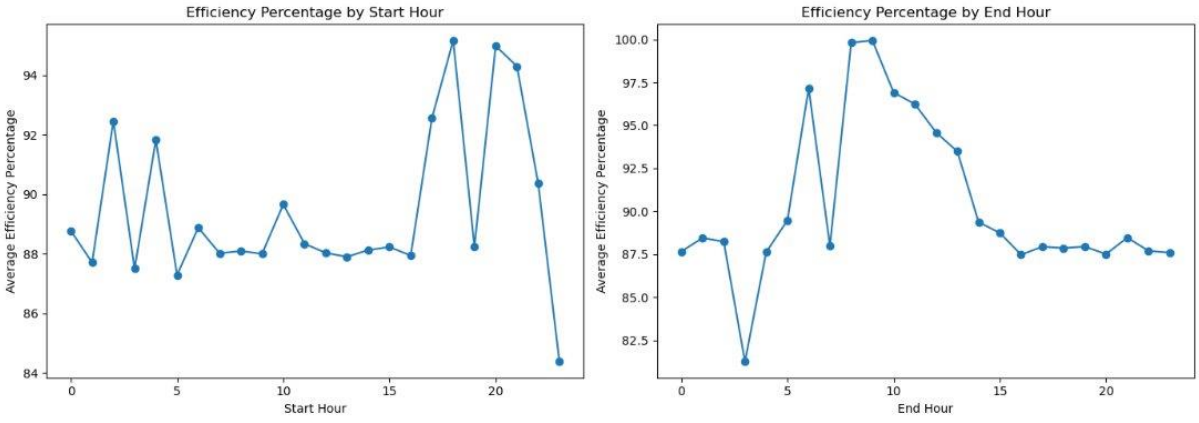

Fig 10: Analysis of Efficiency Percentage by Start and End Hours

```
import pandas as pd

import matplotlib.pyplot as plt

data['MinOfFrom Time'] = pd.to_datetime(data['MinOfFrom Time']).dt.hour

data['MaxOfTo Time'] = pd.to_datetime(data['MaxOfTo Time']).dt.hour

# Group the data by 'Start Hour' and calculate average efficiency
```

avg_efficiency_by_start = data.groupby('MinOfFrom Time')['Efficiency Percentage'].mean().reset_index()

 # Group the data by 'End Hour' and calculate average efficiency

avg_efficiency_by_end = data.groupby('MaxOfTo Time')['Efficiency Percentage'].mean().reset_index()

 # Plotting the average efficiency by start hour

plt.figure(figsize=(14, 5))

plt.subplot(1, 2, 1)

plt.plot(avg_efficiency_by_start['MinOfFrom Time'], avg_efficiency_by_start['Efficiency Percentage'], marker='o')

plt.title('Efficiency Percentage by Start Hour')

plt.xlabel('Start Hour')

plt.ylabel('Average Efficiency Percentage')

 # Plotting the average efficiency by end hour

plt.subplot(1, 2, 2)

plt.plot(avg_efficiency_by_end['MaxOfTo Time'], avg_efficiency_by_end['Efficiency Percentage'], marker='o')

plt.title('Efficiency Percentage by End Hour')

plt.xlabel('End Hour')

plt.ylabel('Average Efficiency Percentage')

plt.tight_layout()

plt.show()

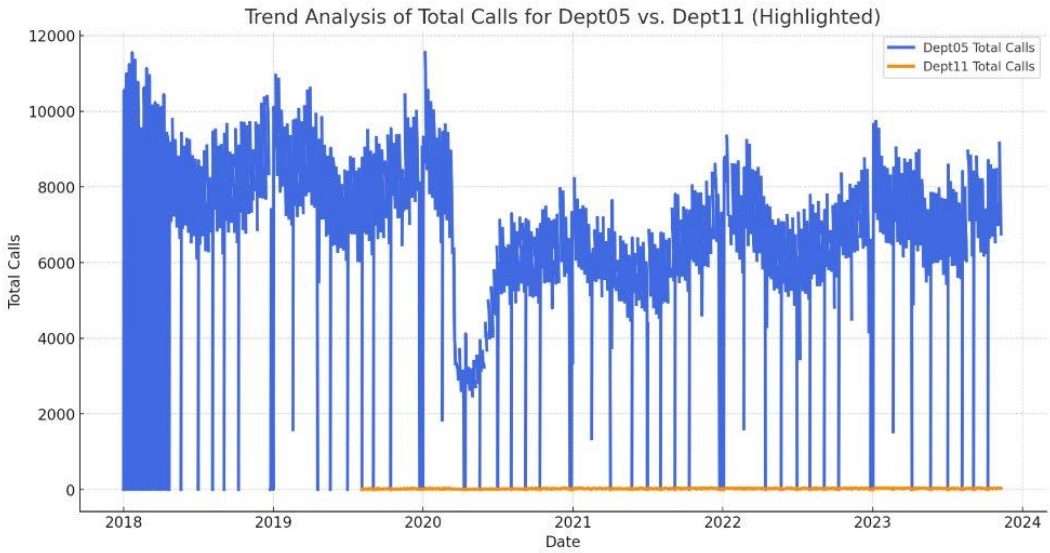

Fig 11: Comparative Call Volume Trend Between Department 05 and Department 11

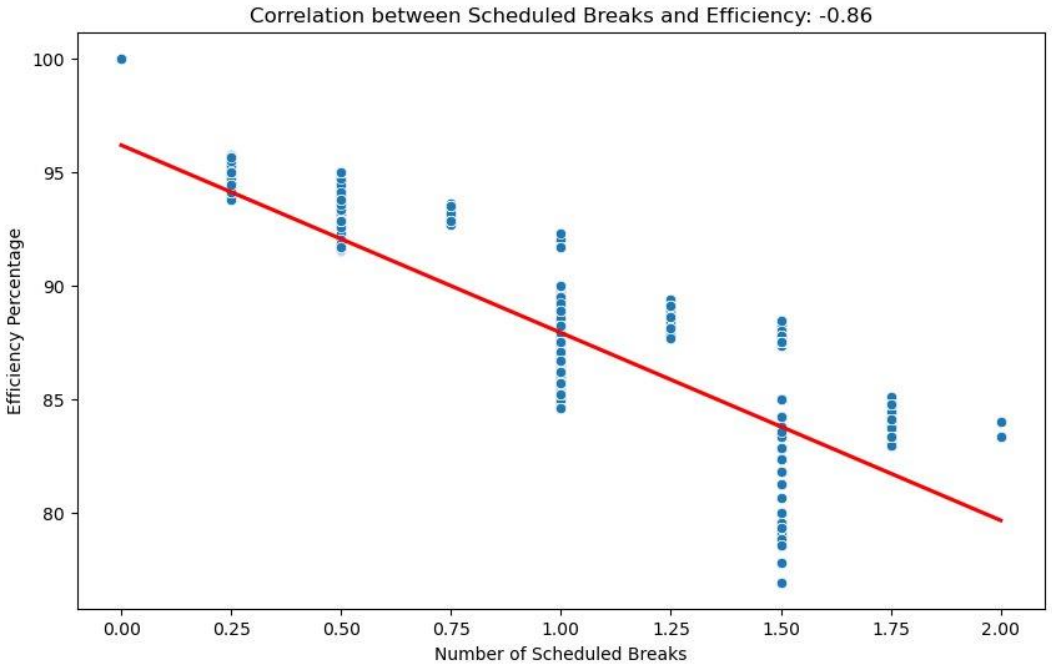

Fig 12: Scatter Plot of Efficiency Percentage vs. Number of Scheduled Breaks

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
# Calculate the correlation coefficient between scheduled breaks and efficiency
correlation = df['SchBreaks'].corr(df['Efficiency Percentage'])
# Plot the relationship between scheduled breaks and efficiency
plt.figure(figsize=(10, 6))
sns.scatterplot(x='SchBreaks', y='Efficiency Percentage', data=df)
sns.regplot(x='SchBreaks', y='Efficiency Percentage', data=df, scatter=False, color='red')
plt.title(f'Correlation between Scheduled Breaks and Efficiency: {correlation:.2f}')
plt.xlabel('Number of Scheduled Breaks')
plt.ylabel('Efficiency Percentage')
plt.show()
```

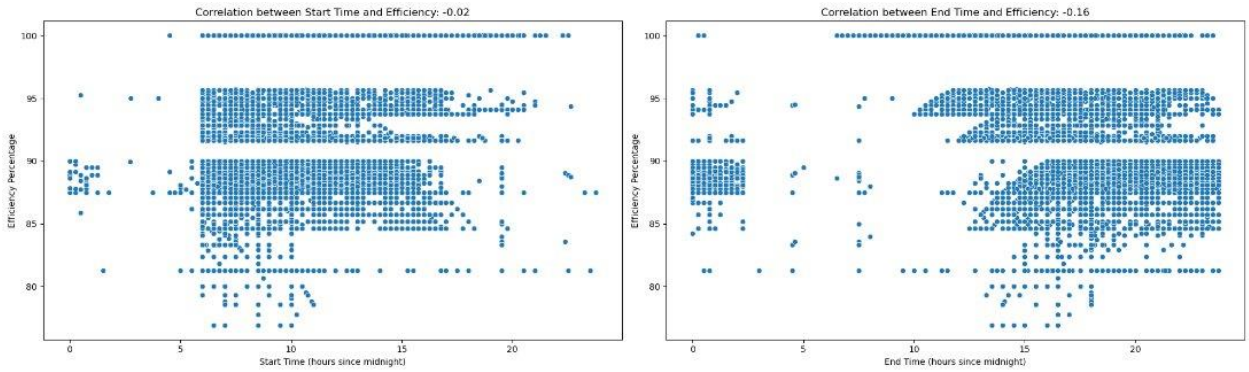

Fig 13: Efficiency Correlation with Start and End Times

```
data = pd.read_excel('C:/Users/Sreeja/OneDrive/Desktop/PMD/Production1.xlsx')

data['Start Time Numeric'] = data['MinOfFrom Time'].dt.hour + data['MinOfFrom Time'].dt.minute / 60

data['End Time Numeric'] = data['MaxOfTo Time'].dt.hour + data['MaxOfTo Time'].dt.minute / 60

# Calculate the correlation between start times, end times, and efficiency

correlation_start_efficiency = data['Start Time Numeric'].corr(data['Efficiency Percentage'])

correlation_end_efficiency = data['End Time Numeric'].corr(data['Efficiency Percentage'])

# Create scatter plots to visualize these relationships

plt.figure(figsize=(20, 6))

# Start Time vs Efficiency

plt.subplot(1, 2, 1)

sns.scatterplot(x='Start Time Numeric', y='Efficiency Percentage', data=data)

plt.title(f'Correlation between Start Time and Efficiency: {correlation_start_efficiency:.2f}')

plt.xlabel('Start Time (hours since midnight)')

plt.ylabel('Efficiency Percentage')

# End Time vs Efficiency

plt.subplot(1, 2, 2)

sns.scatterplot(x='End Time Numeric', y='Efficiency Percentage', data=data)

plt.title(f'Correlation between End Time and Efficiency: {correlation_end_efficiency:.2f}')

plt.xlabel('End Time (hours since midnight)')

plt.ylabel('Efficiency Percentage')

plt.tight_layout()

plt.show()
```

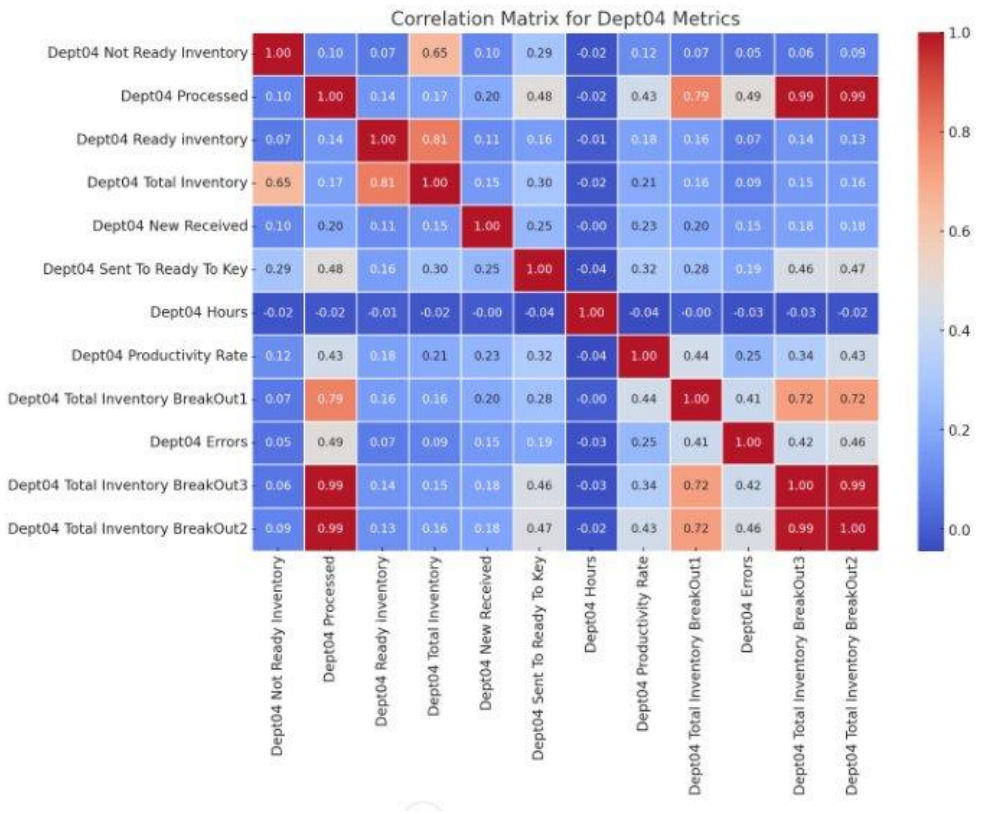

Fig 14: Heatmap of Correlation Coefficients for Department 04 Operational Metrics

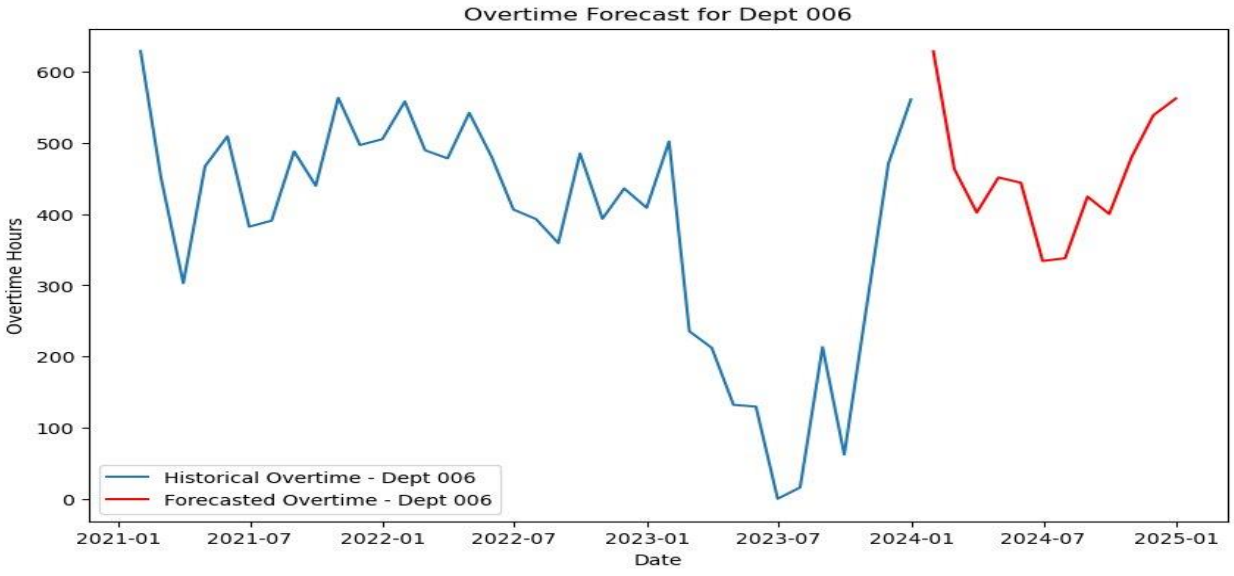

Fig 15: Historical vs. Forecasted Overtime Hours for Department 006

```
import pandas as pd
import numpy as np
from statsmodels.tsa.statespace.sarimax import SARIMAX
import matplotlib.pyplot as plt
df['Date'] = pd.to_datetime(df['Date'])
```

df.set_index('Date', inplace=True)

 # Filter the dataset for "Dept 006" only

df_dept006 = df[df['Dept ID'] == 'Dept006']

 # Aggregate data to get total overtime hours by month specifically for "Dept 006"

monthly_overtime_dept006 = df_dept006['OT Portion'].resample('M').sum()

 # Define the SARIMA model; adjust the order and seasonal_order as necessary

sarima_model = SARIMAX(monthly_overtime_dept006, order=(1, 1, 1), seasonal_order=(1, 1, 1, 12))

 # Fit the model

sarima_result = sarima_model.fit(disp=False)

 # Make forecast for the next 12 periods (e.g., 12 months)

forecast = sarima_result.get_forecast(steps=12)

 # Plot the past overtime data and the forecast with confidence intervals for "Dept 006"

plt.figure(figsize=(10, 6))

plt.plot(monthly_overtime_dept006.index, monthly_overtime_dept006, label='Historical Overtime - Dept 006')

plt.plot(forecast.predicted_mean.index, forecast.predicted_mean, label='Forecasted Overtime - Dept 006', color='red')

plt.title('Overtime Forecast for Dept 006')

plt.xlabel('Date')

plt.ylabel('Overtime Hours')

plt.legend()

plt.show()

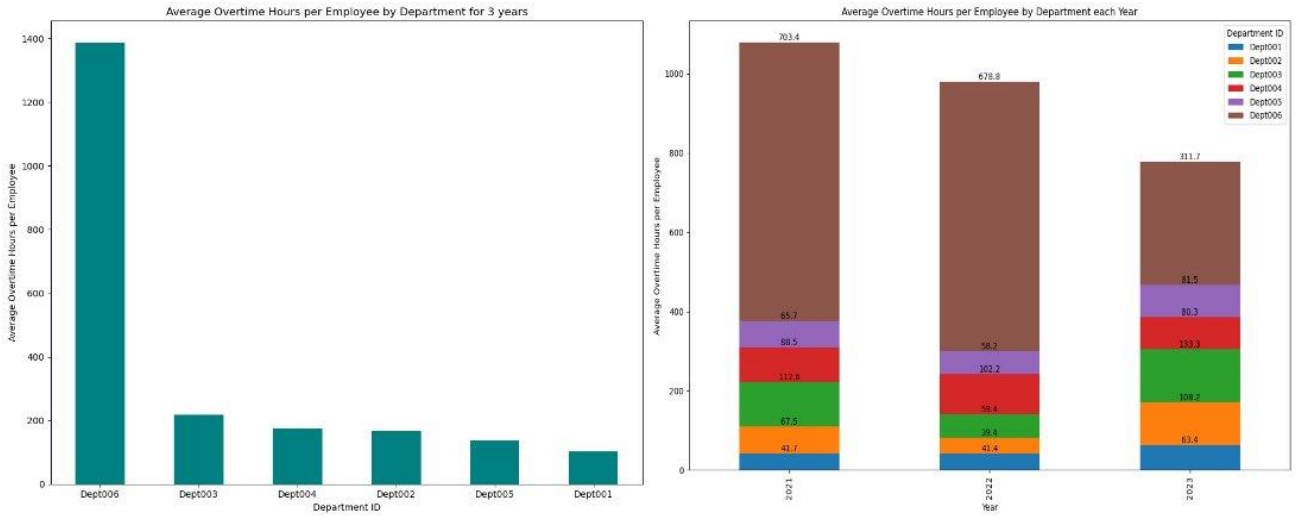

Fig 16: Average Overtime Hours per Employee by Department Over 3 Years

And Yearly Breakdown of Average Overtime Hours per Employee by Department

```
import pandas as pd
import matplotlib.pyplot as plt
# Calculate the total overtime hours per department
total_overtime_per_dept = df.groupby('Dept ID')['OT Portion'].sum()
# Calculate the number of employees in each department
employee_count_per_dept = df.groupby('Dept ID')['EmployeeID'].nunique()
# Calculate the average overtime per employee for each department
avg_overtime_per_employee = total_overtime_per_dept / employee_count_per_dept
# Sort the departments by average overtime per employee for better visualization
avg_overtime_per_employee_sorted = avg_overtime_per_employee.sort_values(ascending=False)
# Plotting
plt.figure(figsize=(10, 8))
avg_overtime_per_employee_sorted.plot(kind='bar', color='teal')
plt.title('Average Overtime Hours per Employee by Department for 3 years')
plt.xlabel('Department ID')
plt.ylabel('Average Overtime Hours per Employee')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```

Yearly basis

```
# Extract the year from the 'Date' column
df['Year'] = pd.DatetimeIndex(df['Date']).year
# Group by Department and Year to calculate total overtime hours
total_overtime_per_dept_year = df.groupby(['Dept ID', 'Year'])['OT Portion'].sum()
# Calculate the number of unique employees in each department per year
employee_count_per_dept_year = df.groupby(['Dept ID', 'Year'])['EmployeeID'].nunique()
# Calculate the average overtime per employee for each department per year
avg_overtime_per_employee_per_dept_year = total_overtime_per_dept_year /
employee_count_per_dept_year
# Reset the index to turn the multi-index into columns
avg_overtime_per_employee_per_dept_year =
avg_overtime_per_employee_per_dept_year.reset_index()
# Pivot the data to prepare for plotting
```

```
pivot_df = avg_overtime_per_employee_per_dept_year.pivot(index='Year', columns='Dept ID', values=0)

# Create a bar plot for the average overtime per employee for each department per year

ax = pivot_df.plot(kind='bar', figsize=(12, 8), stacked=True)

plt.title('Average Overtime Hours per Employee by Department each Year')

plt.xlabel('Year')

plt.ylabel('Average Overtime Hours per Employee')

plt.legend(title='Department ID')

plt.tight_layout()

# Adding the numbers on top of the bars

for p in ax.patches:

    width = p.get_width()

    height = p.get_height()

    x, y = p.get_xy()

    if height > 0:

        ax.text(x+width/2,

                y+height,

                '{:.1f}'.format(height),

                ha='center',

                va='bottom')

plt.show()
```

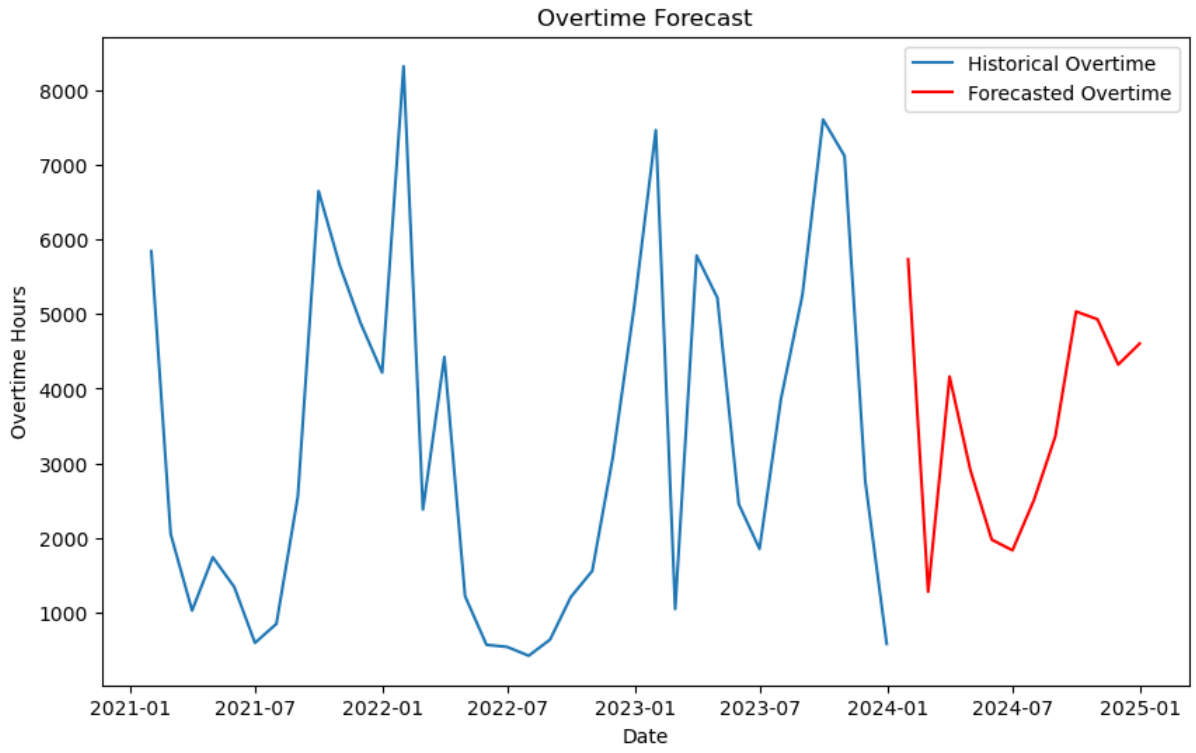

Fig 17: Historical and Projected Overtime Hours Trend

```
import pandas as pd
import numpy as np
from statsmodels.tsa.statespace.sarimax import SARIMAX
import matplotlib.pyplot as plt
df['Date'] = pd.to_datetime(df['Date'])
df.set_index('Date', inplace=True)
monthly_overtime = df['OT Portion'].resample('M').sum()
sarima_model = SARIMAX(monthly_overtime, order=(1, 1, 1), seasonal_order=(1, 1, 1, 12))
# Fit the model
sarima_result = sarima_model.fit(disp=False)
forecast = sarima_result.get_forecast(steps=12)
# Plot the past overtime data and the forecast with confidence intervals
plt.figure(figsize=(10, 6))
plt.plot(monthly_overtime.index, monthly_overtime, label='Historical Overtime')
plt.plot(forecast.predicted_mean.index, forecast.predicted_mean, label='Forecasted Overtime',
color='red')
plt.title('Overtime Forecast')
plt.xlabel('Date')
```

plt.ylabel('Overtime Hours')

plt.legend()

plt.show()

from sklearn.metrics import mean_absolute_error, mean_squared_error

 # Calculate MAE

mae = mean_absolute_error(actuals, forecast.predicted_mean)

 # Calculate RMSE

rmse = np.sqrt(mean_squared_error(actuals, forecast.predicted_mean))

 # Calculate MAPE - Handle division by zero if it occurs in actuals

*mape = np.mean(np.abs((actuals - forecast.predicted_mean) / actuals)) * 100 if not (actuals == 0).any() else np.inf*

 # Print the error metrics

print(f"MAE: {mae}")

print(f"RMSE: {rmse}")

print(f"MAPE: {mape}%")

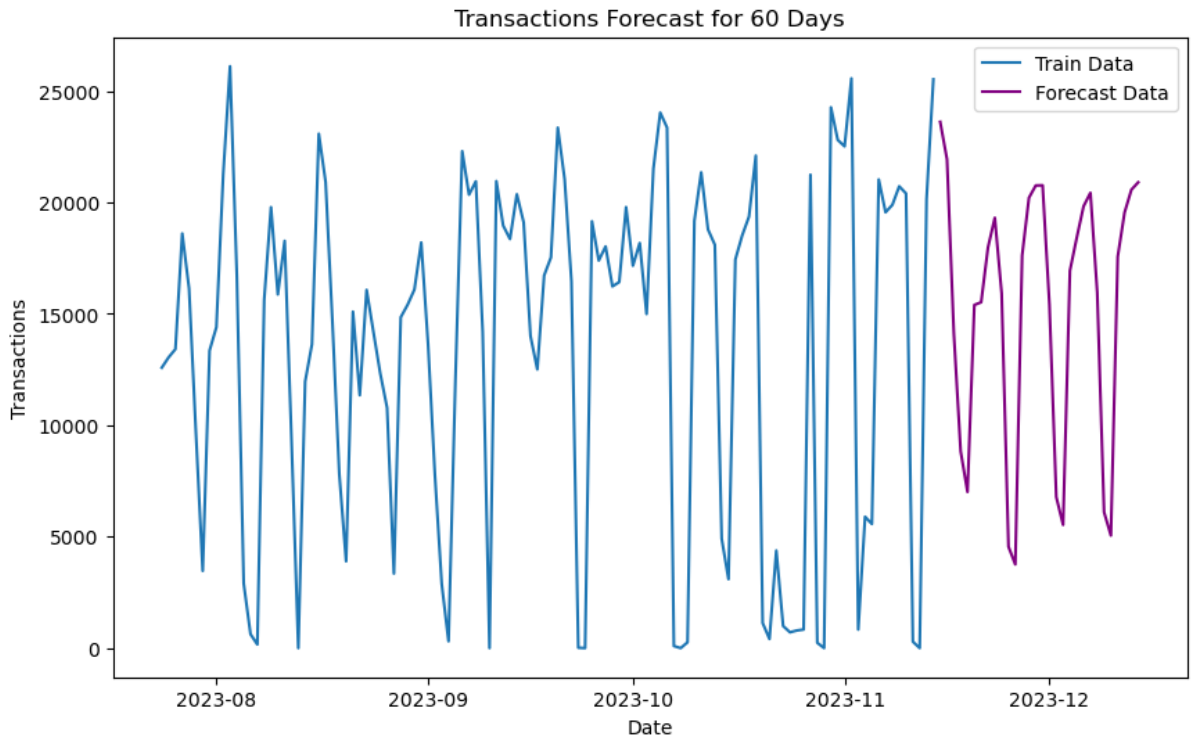

Fig 18: Transactions forecast

import pandas as pd

from statsmodels.tsa.statespace.sarimax import SARIMAX

```
import matplotlib.pyplot as plt

transactions_data['DateofShiftStart'] = pd.to_datetime(transactions_data['DateofShiftStart'])

transactions_data.set_index('DateofShiftStart', inplace=True)

daily_transactions = transactions_data['Transactions'].resample('D').sum()

daily_transactions.fillna(0, inplace=True)

train_data = daily_transactions

sarima_order = (1, 1, 1)

seasonal_order = (1, 1, 1, 7)

# Fit the SARIMA model

model = SARIMAX(train_data, order=sarima_order, seasonal_order=seasonal_order)

results = model.fit()

# Forecast the next 60 days

forecast = results.get_forecast(steps=30)

forecast_index = pd.date_range(start=train_data.index[-1], periods=31, freq='D')[1:]  # Generate future dates for 60 days

# Create a DataFrame for the 60-day forecast

forecast_df = pd.DataFrame({'Forecast': forecast.predicted_mean}, index=forecast_index)

# Plot the actual data and forecasts

plt.figure(figsize=(10, 6))

plt.plot(train_data, label='Train Data')

plt.plot(forecast_df, label='Forecast Data', color='purple')

plt.title('Transactions Forecast for 60 Days')

plt.xlabel('Date')

plt.ylabel('Transactions')

plt.legend()
```

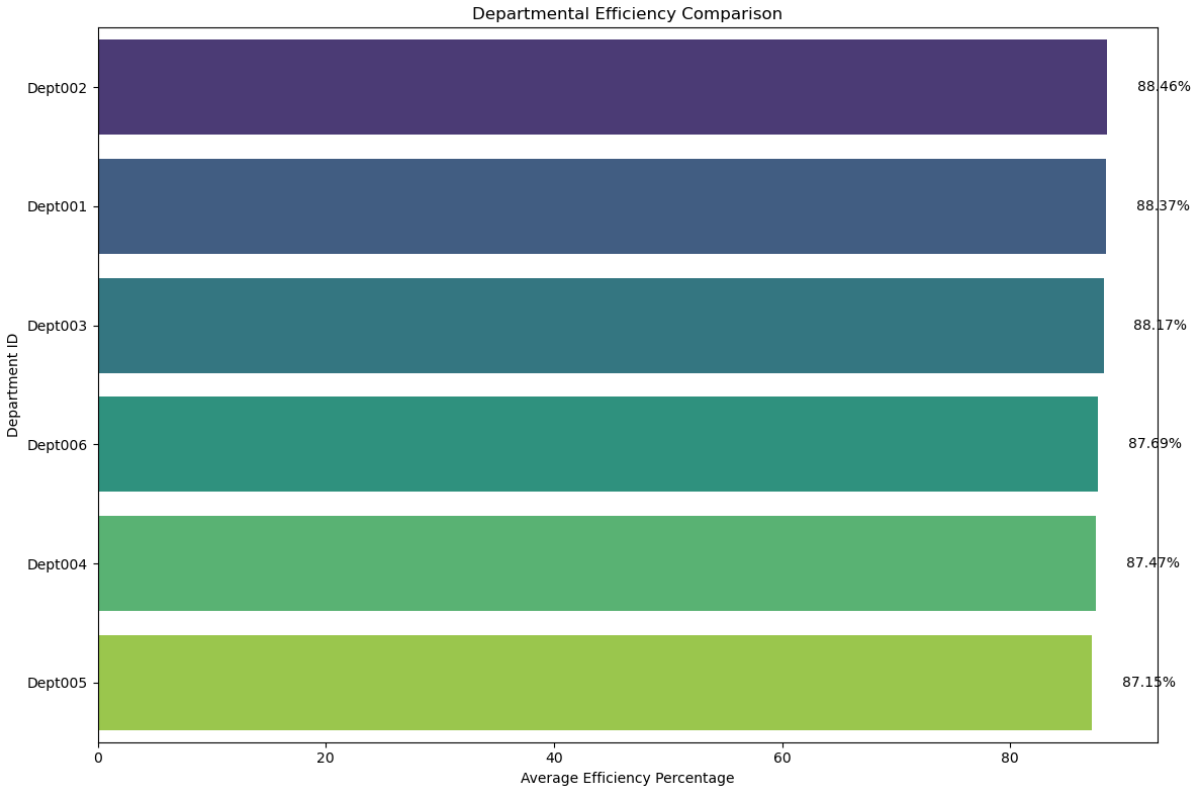

Fig 19. Efficiency of Departments

```
import matplotlib.pyplot as plt

import seaborn as sns

import pandas as pd

department_efficiency = data.groupby('Dept ID')['Efficiency Percentage'].mean().reset_index()

department_efficiency_sorted = department_efficiency.sort_values('Efficiency Percentage', ascending=False)

# Visualization

plt.figure(figsize=(12, 8))

bar_plot = sns.barplot(x='Efficiency Percentage', y='Dept ID', data=department_efficiency_sorted, palette='viridis')

plt.title('Departmental Efficiency Comparison')

plt.xlabel('Average Efficiency Percentage')

plt.ylabel('Department ID')

# Adding annotations to each bar

for p in bar_plot.patches:

    width = p.get_width()

    plt.text(5 + width, p.get_y() + p.get_height() / 2,

             '{:1.2f}%'.format(width),
```

```
        ha='center', va='center')
plt.tight_layout()
# Display the plot
plt.show()
```