Moataz Mansour
Senior Database Architect
Swiss Post
Moataz.mansour@bluewin.ch

Tobias Böni
Senior Data Engineer
Tobias.boeni@bluewin.ch

CAS ADS Uni Bern Data Science Final Project

Swiss Post Sorting Center Package Sorting Performance Analysis and Prediction

Final Project Report

15 June 2025

Moataz Mansour/Tobias Böni, CAS Applied Data Science, University of Bern

Abstract

Swiss Post processes a substantial volume of packages each year, delivering around 200 million parcels in 2023 alone. This highlights its crucial role in Switzerland's logistics network, covering both domestic and international shipments. However, in 2023, Swiss Post encountered performance challenges, driven by external factors such as inflation, weakened consumer confidence, and geopolitical events. These factors led to a 4.7% decline in parcel volumes compared to 2022. Despite this, Swiss Post has made considerable investments in its logistics network to address efficiency issues, including the introduction of new regional parcel centers and upgraded sorting technology. These improvements have eased the pressure on processing times in high-demand areas like Zurich and Basel.

The surge in package deliveries that began during the COVID-19 crisis overwhelmed the existing infrastructure, prompting Swiss Post to invest millions in new sorting centers to meet the growing demand. By enhancing the efficiency of the existing sorting centers, Swiss Post could fully optimize their usage, reducing the need for further costly investments in additional facilities.

This project focuses on analyzing sorting center operations performance to identify congestion points and their causes, while also developing predictive models to foresee and prevent performance issues before they occur. This proactive strategy will help improve the overall performance of Swiss Post's sorting centers.

Table of Contents

1. Introduction

The goal of this project is to analyze the postal sorting center's performance by identifying the most influential factors contributing to sorting issues, including shipment attributes (e.g., dimensions, weight, coding stations, and timestamps) as well as chute congestion which we believe have a major impact on overall performance of the center and finally using a model to determine whether overburdened chutes or certain features create bottlenecks that reduce overall system efficiency.

The sorting process consists of three key stages:

A: Shipments **arrive** and are delivered to the sorting center, using designated units

B: Shipments are **scanned and transferred** to the conveyor belts, where automatic and manual sorting machines determine their route and send them to the appropriate chute based on the destination.

C: The **chute serves as the output** of the sorting machine, directing the parcels to different destinations depending on the ZIP code., one chute can serve several ZIP codes

The center's performance is measured by the number of parcels processed per time frame. Some centers have shown up to a 15% increase in processing efficiency compared to others with similar setups. Further investigation revealed that traffic bottlenecks at certain chutes, which handle significantly higher package volumes, cause an imbalance. This results in a non-normal distribution of packages across the chutes, leading to a noticeable reduction in overall performance.

Our goal is to achieve a balanced, normally distributed flow of packages across all available chutes, which would enhance the sorting rate and improve overall center performance.

Specific Objectives:

1.1. **Determine Feature Importance**: Rank the shipment features by their importance and identify which shipment attributes (e.g., dimensions, weight, coding station) are most influential in causing sorting issues at postal centers. As well Correlation between the features and the impact on performance

1.2. **Determine chute congestion impact:** Determine whether chutes are handling disproportionately large volumes of packages can create bottlenecks that reduce overall system efficiency

1.3. **Predict Sorting Performance and Issues**: Develop a predictive model capable of forecasting sorting issues based on historical shipment data.

1.4. **Generate Actionable Insights**: Provide data-driven recommendations for improving chute utilization and enhance overall performance

2. Infrastructure and Tools

Swiss Post's IT infrastructure consists of over 5,300 databases and more than 800 custom-built applications, distributed across two highly secure data centers supported by a fiber network backbone. These data centers are designed for disaster recovery, ensuring operational continuity. At the core of this system is the Oracle Exadata Cloud at Customer (EXACC) platform, a high-performance database server valued at 5 million CHF, which hosts many of Swiss Post's databases and is mirrored across both data centers for redundancy and high availability in case of system failures.

A- Databases are used to store shipment data and provide real-time updates, ensuring accurate and up-to-date information flow. These systems are built with high availability and redundancy for continuous operation, and they support advanced data analytics and reporting for performance monitoring and decision-making.

B- Applications, including parcel tracking systems, sorting system management, and predictive analytics for forecasting performance issues. Middleware and API integration enable seamless communication between different systems. These applications are hosted on virtualized or containerized platforms (e.g., Docker, Kubernetes), allowing Swiss Post to easily scale its IT resources based on demand.

C- network infrastructure, consisting of high-speed fiber optic networks, redundant architecture with failover mechanisms, VPNs, and multi-cloud integration, ensures reliable connectivity between sorting centers and offices, while stringent security measures, including firewalls, encryption, and regular audits, protect data and maintain compliance.

D- advanced control and monitoring systems for its sorting centers, such as SCADA (Supervisory Control and Data Acquisition) systems to oversee sorting operations, and centralized command centers that manage and coordinate the activities of the sorting centers.

E- sorting centers in Härkingen, Frauenfeld, Daillens, Wallisellen (new), and Pratteln (new) are equipped with automated sorting machines connected to the IT infrastructure through IoT sensors. These sensors provide continuous data streams to monitor sorting accuracy, operational efficiency, and package flow in real time.

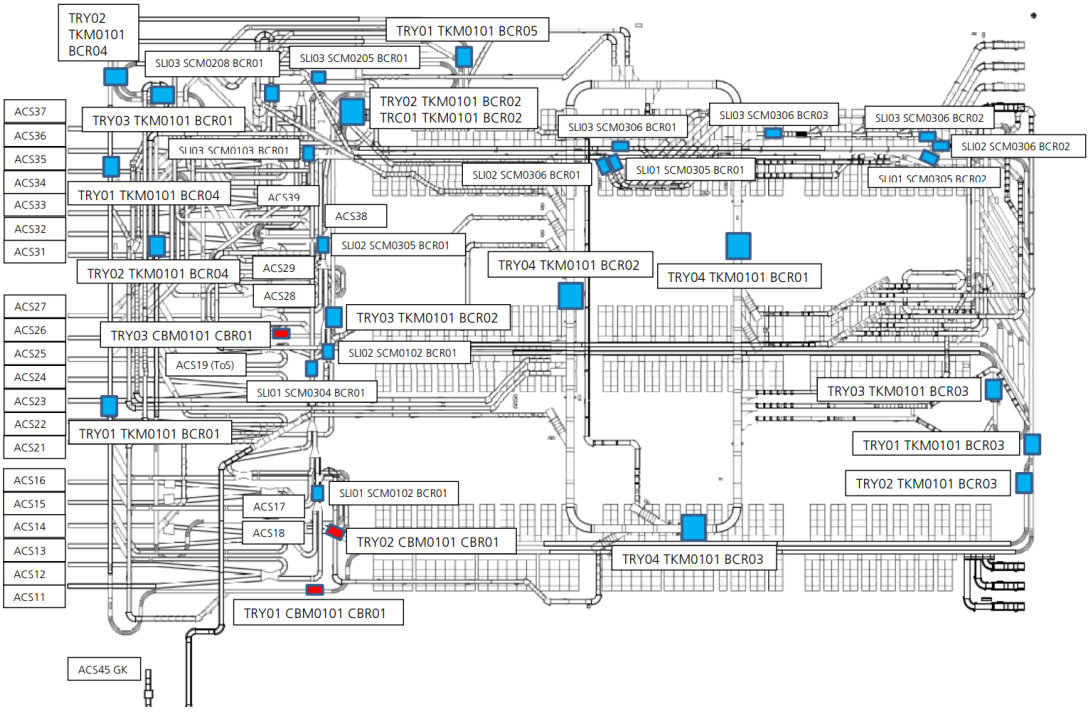
Swiss Post's comprehensive IT infrastructure, featuring powerful databases, application servers, and a robust network architecture, serves as the backbone of its logistics operations, enabling seamless parcel processing and tracking nationwide. With its scalability, security, and redundancy, the system ensures reliable and efficient performance, even during peak demand periods.

- ## infrastructure and tools used for this project:

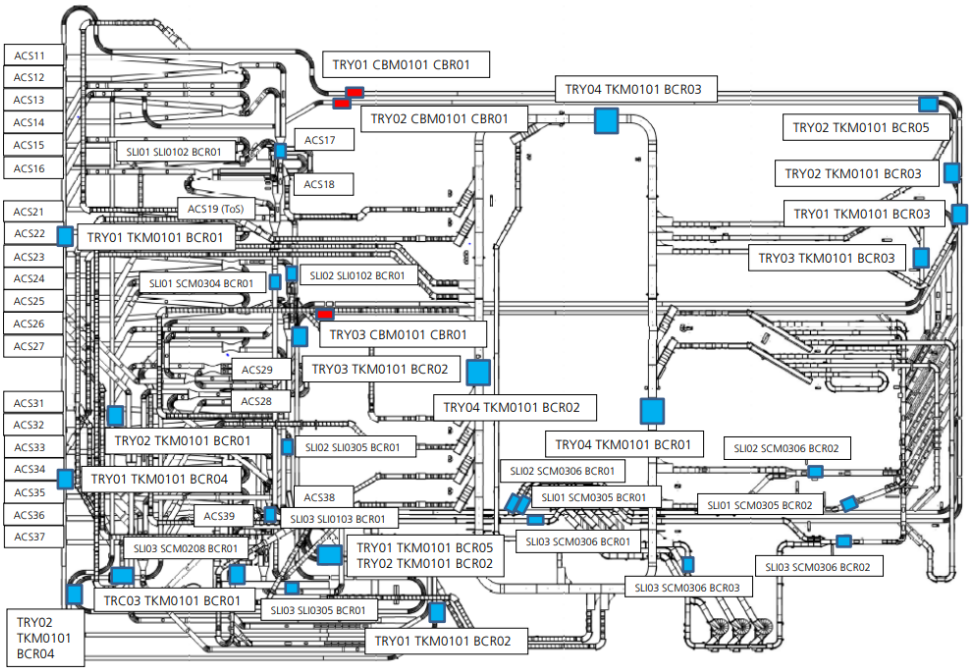
We have limited our study to focus on two parcel centers: **Härkingen** and **Frauenfeld**, and are analyzing data from a **single major database** that holds parcel processing information. This database contains massive tables, with an average of **5 billion records**, providing extensive data for our analysis of parcel operations.

- **Database Tools: Oracle 23ai** with vector search and other DB tools and Toad is used to manage the Data extraction and query and cleanup
- **Hardware:** on prem Exacc Database server with huge database power where package databases (PADASA and X) with 50 TB storage usage and 5 billion records tables runs and the Swiss post AWS server cloud infrastructure to run the developed predictive models
- **Python Libraries**:
    - **OracleDB:** using Python SQL To extract relevant parcel data from databases.
    - **Core Libraries:** pandas & numpy: data loading and manipulation, operations and arrays
    - **Visualization:** matplotlib & seaborn : data visualization & Advanced visualizations
    - **Deep Learning : PyTorch :** torch – PyTorch base torch.nn – for defining neural networks (e.g., LSTM, MLP) and torch.utils.data – for datasets and data loaders
    - **Machine Learning & Preprocessing:** sklearn.preprocessing.MinMaxScaler – feature scaling
    - sklearn.metrics – evaluation metrics, mean_absolute_error, accuracy_score, precision_score
    - recall_score, f1_score
    - **Reinforcement Learning:** gym – environment wrapper for RL agents
    - stable_baselines3 – RL agent training (PPO, DQN, etc.)
    - shimmy – compatibility layer for gym + stable_baselines3

Übersicht alle Lesesysteme PZ Härkingen Stand 2022



Übersicht alle Lesesysteme PZ Frauenfeld Stand 2022

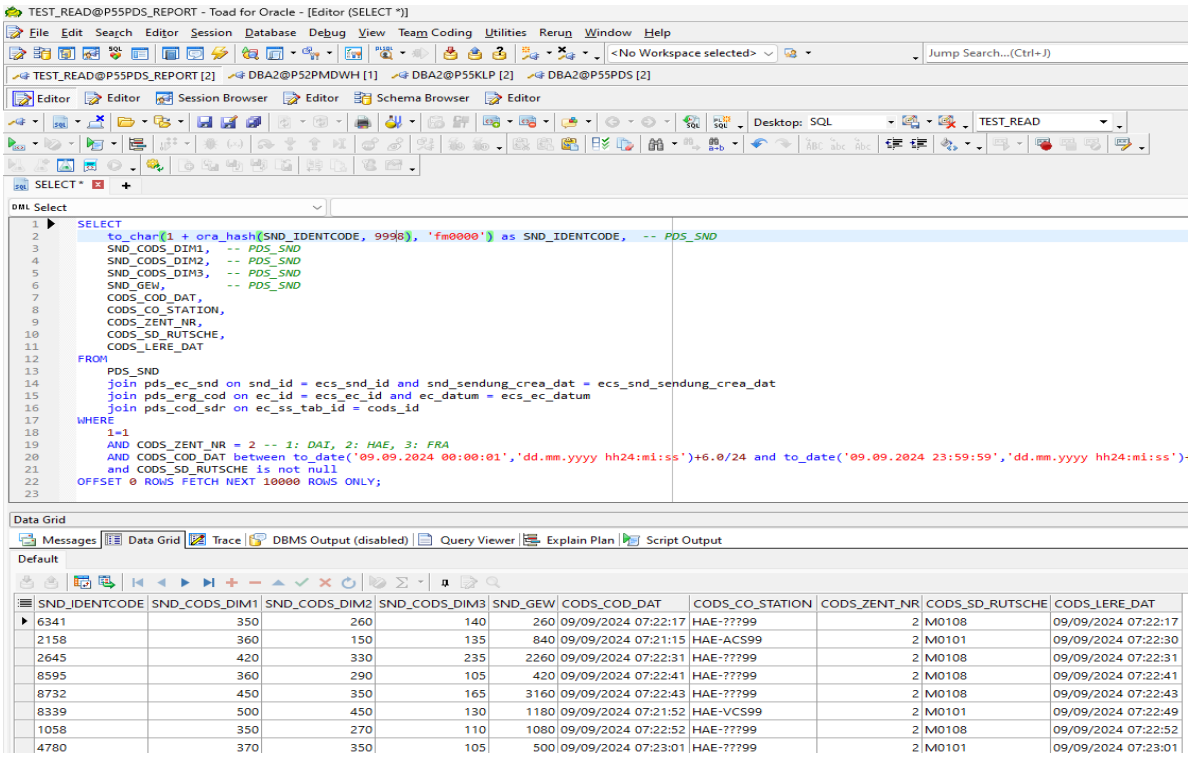


3. DATA

The data for this project comes from Swiss Post's shipment sorting system and includes:

- **Shipment number** (anonymized) SND_IDENTCODE
- **Shipment dimensions**: Length, width, and height (in millimeters) SND_CODS_DIM1, SND_CODS_DIM2, SND_CODS_DIM3
- **Shipment weight**: (n grams) SND_GEW
- **Scanning timestamps** when the item first scanned in the sorting center CODS_COD_DAT
- **Scanner station**: Sorting station identifier CODS_CO_STATION
- **Sorting center Number** CODS_ZENT_NR_x
- **leaving timestamps** when the item left the sorting center chute CODS_LERE_DAT
- **chute station** where the item is sent CODS_SD_RUTSCHE

3.1. Sample Data Example



SND_IDENTCODE	SND_CODS_DIM1	SND_CODS_DIM2	SND_CODS_DIM3	SND_GEW	CODS_COD_DAT	CODS_CO_STATION	CODS_ZENT_NR	CODS_SD_RUTSCHE	CODS_LERE_DAT
6341	350	260	140	260	09/09/2024 07:22:17	HAE-???99	2	M0108	09/09/2024 07:22:17
2158	360	150	135	840	09/09/2024 07:21:15	HAE-ACS99	2	M0101	09/09/2024 07:22:30
2645	420	330	235	2260	09/09/2024 07:22:31	HAE-???99	2	M0108	09/09/2024 07:22:31
8595	360	290	105	420	09/09/2024 07:22:41	HAE-???99	2	M0108	09/09/2024 07:22:41
8732	450	350	165	3160	09/09/2024 07:22:43	HAE-???99	2	M0108	09/09/2024 07:22:43
8339	500	450	130	1180	09/09/2024 07:21:52	HAE-VCS99	2	M0101	09/09/2024 07:22:49
1058	350	270	110	1080	09/09/2024 07:22:52	HAE-???99	2	M0108	09/09/2024 07:22:52
4780	370	350	105	500	09/09/2024 07:23:01	HAE-???99	2	M0101	09/09/2024 07:23:01

	CODS_IDENTCODE_AN	CODS_DIM1	CODS_DIM2	CODS_DIM3	CODS_GEW	CODS_ADR_PLZZZ_AN	CODS_CO_STATION	CODS_ZENT_NR	CODS_SD_RUTSCHE	DAY_15MIN	CODS_LERE_DAT	MINUTE_COUNTER	PROCESSING_TIME_MINUTES	SORTING_PERF_ISSUE
0	242455661	700	454	54	1480	449561	FRA-ACS18	3	R0109	202411061015	06/11/2024 10:26:25	447026	1	0
1	482169492	704	454	48	1520	568172	FRA-ACS18	3	R0109	202411061015	06/11/2024 10:26:15	447026	2	0
2	407572193	400	380	235	4700	23223	FRA-ACS21	3	R2515	202411061015	06/11/2024 10:26:16	447026	4	0
3	999133381	470	450	120	1060	629595	FRA-ACS21	3	R0109	202411061015	06/11/2024 10:26:04	447026	7	0
4	748472889	360	290	110	1480	421619	FRA-ACS21	3	R2319	202411061015	06/11/2024 10:26:37	447027	15	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
170012	302468774	490	390	90	1280	712077	DAI-ACS13	1	R2419	202411061430	06/11/2024 14:38:26	447278	4	0
170013	225629123	170	110	55	120	869319	DAI-ACS13	1	R1223	202411061430	06/11/2024 14:38:22	447278	2	0
170014	822331195	410	360	55	820	49468	DAI-ACS13	1	R2704	202411061430	06/11/2024 14:38:25	447278	6	0
170015	804061373	400	310	110	1640	32705	DAI-ACS13	1	R1709	202411061430	06/11/2024 14:38:19	447278	6	0
170016	486028131	410	300	110	2900	608088	DAI-ACS13	1	R2207	202411061430	06/11/2024 14:38:32	447279	2	0

809022 rows × 14 columns

3.2. Security Considerations

- **Anonymization**: All shipment numbers are anonymized to protect sensitive data.
- **Data Storage**: Data is stored on secure servers and access is controlled according to Swiss Post's privacy and security guidelines.

3.3. Required Metadata

Metadata such as the coding station, shipment size, and coding timestamps are critical for reproducing the analysis. These attributes allow for the recreation of sorting scenarios and the identification of problematic shipments.

Field Name	Description	Data Type	Example
SND_IDENTCODE	Unique identifier for each shipment (anonymized for privacy)	String	A12345
SND_CODS_DIM1	Length of the shipment in millimeters	Integer	300
SND_CODS_DIM2	Width of the shipment in millimeters	Integer	150
SND_CODS_DIM3	Height of the shipment in millimeters	Integer	50
SND_GEW	Weight of the shipment in grams	Integer	1000
CODS_COD_DAT	Timestamp indicating when the shipment was scanned into the sorting center	Datetime	15.01.2023 08:32
CODS_LERE_DAT	Timestamp indicating when the shipment left the sorting center	Datetime	15.01.2023 09:45
CODS_CO_STATION	Station or scanner ID at which the shipment was processed	String	STATION01
CODS_SD_RUTSCHE	Chute identifier where the package was routed for further processing	String	CHUTE10
processing_time_minutes	Calculated field representing the time taken to process a shipment in minutes	Float	73.5

- Dataset Structure: The dataset consists of shipment records, where each row represents a unique shipment with detailed metadata about its dimensions, weight, timestamps, processing station, and chute.

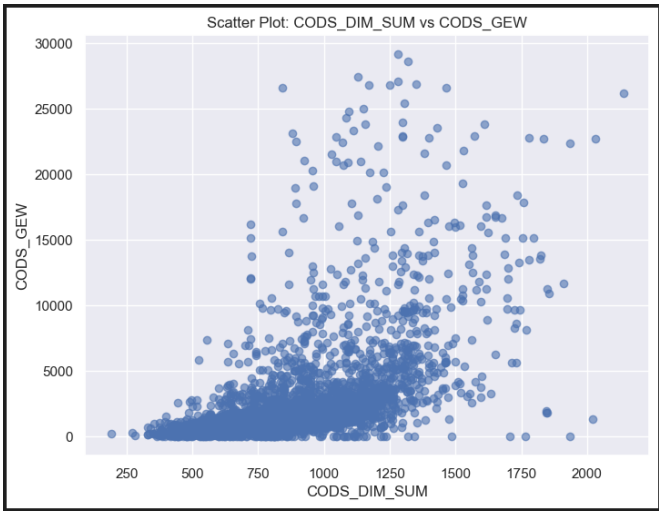
- Calculated Field: processing_time_minutes is derived from CODS_COD_DAT and CODS_LERE_DAT to measure the time a shipment spends in the sorting center.

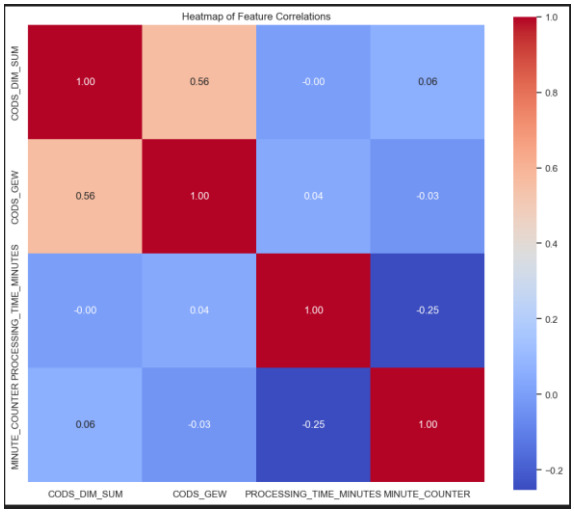
3.4. Metadata Storage

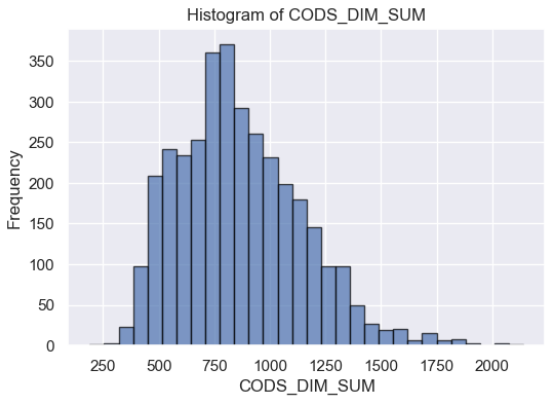
- Metadata will be stored alongside the shipment data in a secure database, with access controlled by Swiss Post.
- Authorized users can access the metadata through SQL queries and data dumps exported to CSV for further analysis.

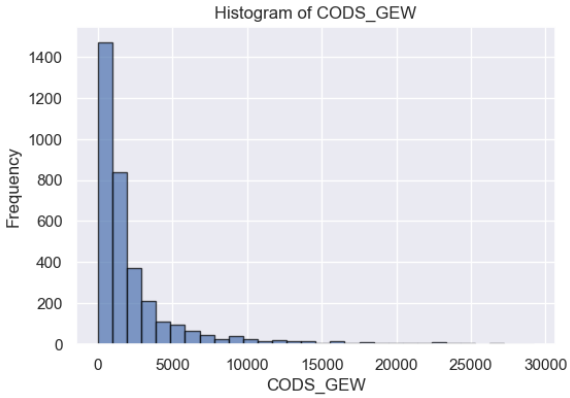
4. Exploratory data analysis (Statistical Descriptive Analysis)

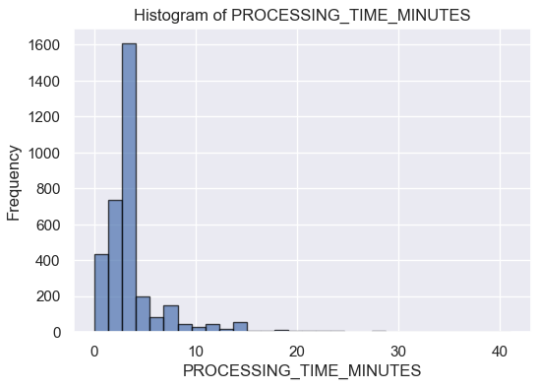
- **Chute Utilization**: The number of packages processed per chute and the average processing time per chute were calculated. This analysis helped identify chutes that were handling a disproportionately large number of shipments and thus were more likely to experience congestion.
- **Performance Metrics**: Center-wide performance metrics were computed, including average processing time per package, total number of packages processed, and individual chute performance.
- **Data Correlation:** The correlation between specific shipments or packages and performance bottlenecks will be analyzed to determine if certain supplier lots arriving at the center are contributing to the overutilization of particular chutes. By identifying these patterns, proactive redistribution measures can be implemented to prevent bottlenecks and optimize center efficiency.

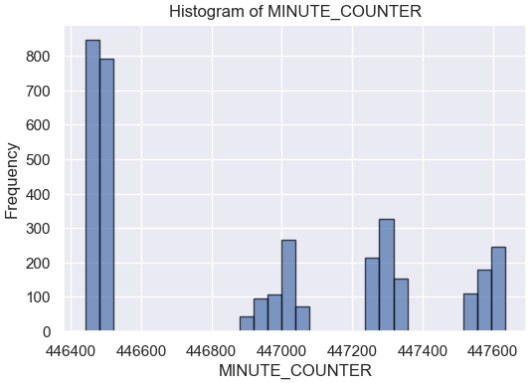


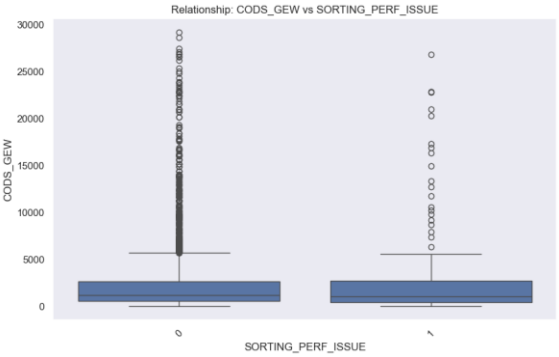


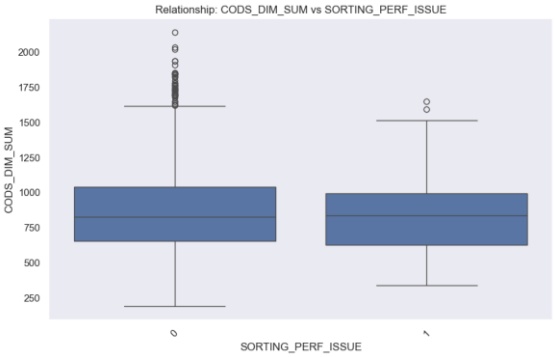

Histogram of CODS_DIM_SUM
Frequency
CODS_DIM_SUM

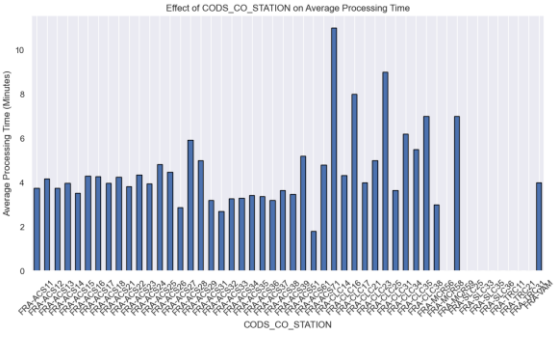

Histogram of CODS_GEW
Frequency
CODS_GEW

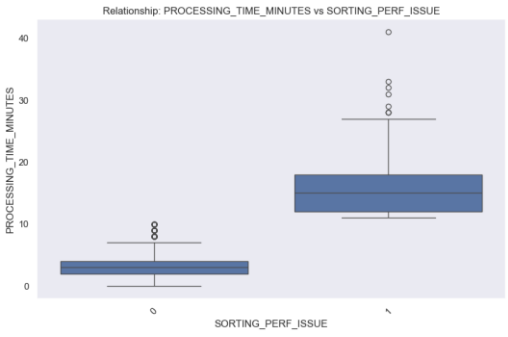

Histogram of PROCESSING_TIME_MINUTES
Frequency
PROCESSING_TIME_MINUTES


Histogram of MINUTE_COUNTER
Frequency
MINUTE_COUNTER

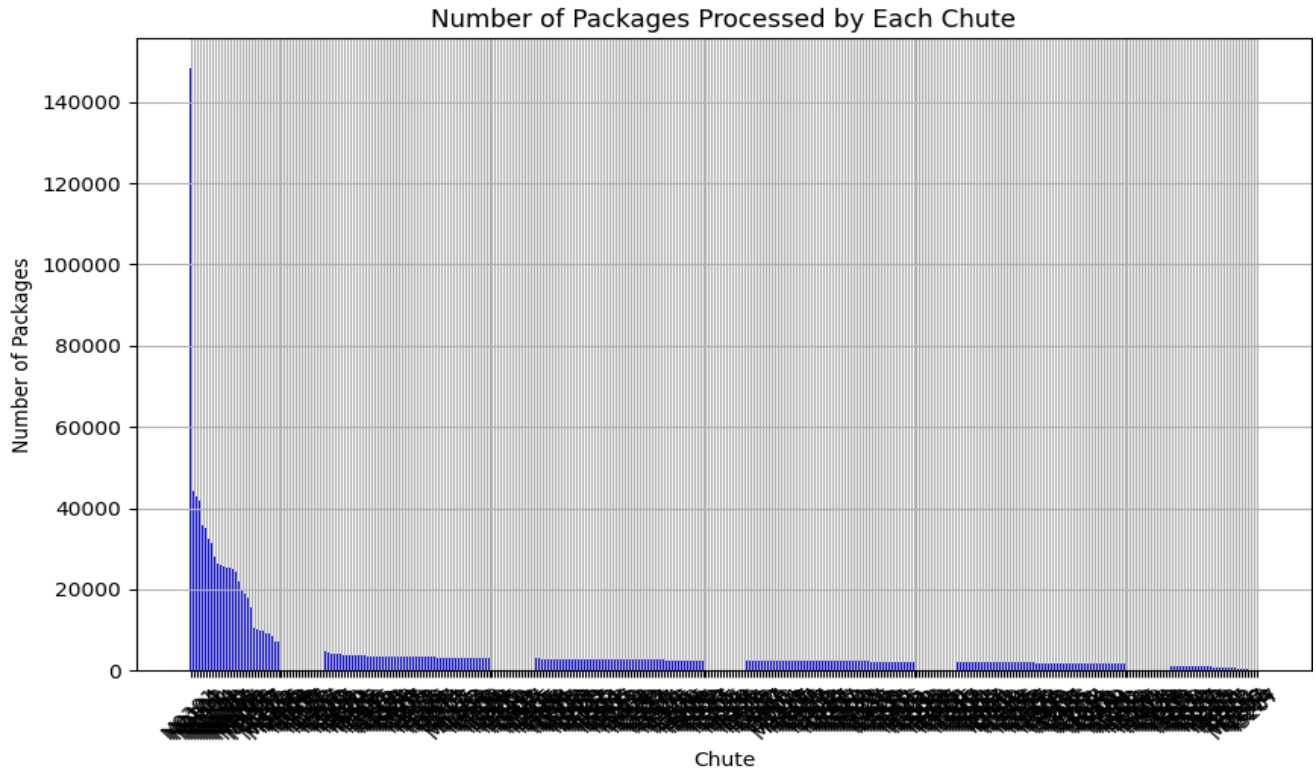

Relationship: CODS_GEW vs SORTING_PERF_ISSUE
CODS_GEW
SORTING_PERF_ISSUE

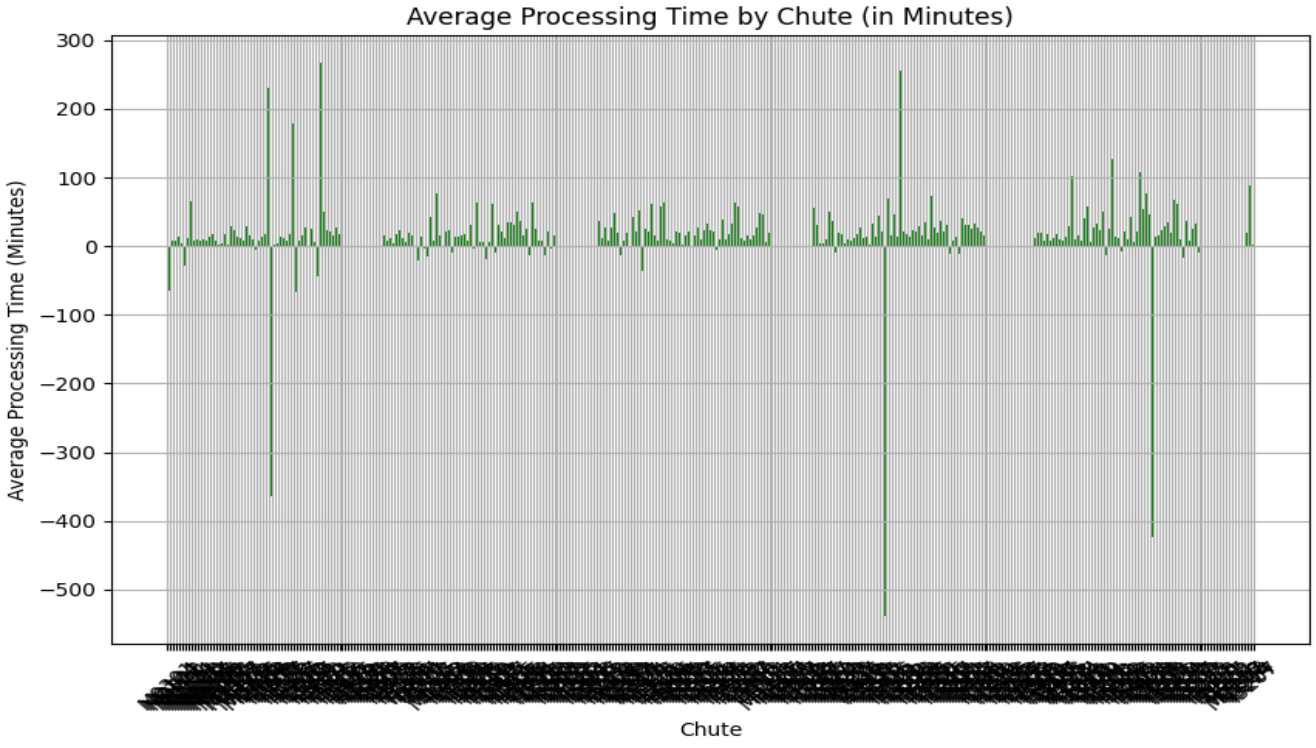

Relationship: CODS_DIM_SUM vs SORTING_PERF_ISSUE
CODS_DIM_SUM
SORTING_PERF_ISSUE


Effect of CODS_CO_STATION on Average Processing Time
Average Processing Time (Minutes)
CODS_CO_STATION


Relationship: PROCESSING_TIME_MINUTES vs SORTING_PERF_ISSUE
PROCESSING_TIME_MINUTES
SORTING_PERF_ISSUE

Histograms and Plots to visualize the distribution of shipment across the Chutes





4.1. Calculating Performance (processing_time_minutes)

To evaluate the performance of the sorting center, we introduced a new column that captures processing time, which is a key performance metric. Processing time refers to the duration a package spends in the sorting process, from the moment it is scanned until it leaves the sorting center, added to the dataset

processing_time_minutes (i) = CODS_COD_DAT(i) - CODS_LERE_DAT(i)

Purpose of the Performance Columns:

a. Identify Delays: The processing_time_minutes column allows us to detect delays and inefficiencies in the sorting process. A high value indicates a potential issue, such as congestion or slow processing, while a lower value suggests efficient performance.

b. Compare Across Stations/Chutes: By grouping the data by sorting stations or chutes, we can compare average processing times and detect whether certain stations or chutes are causing bottlenecks.

c. Overall Center Performance can be calculated on time period bases like daily or hourly and then can be compared with other centers to identify which is more performant and validate the predictions

Performance (Center) = SUM (processing_time_minutes (i)) / Count

4.2. Data Preprocessing

The raw dataset was first preprocessed to ensure that it was suitable for analysis and modeling. Key steps in the preprocessing pipeline included:

- **Data Cleaning**: Missing values in key fields such as CODS_COD_DAT and CODS_LERE_DAT (timestamps) were filtered out. Inconsistent data, such as negative processing times, were also removed.
- **Feature Creation**:The processing_time_minutes column was derived by calculating the difference between the entry and exit timestamps for each shipment.
- Additional features, such as the number of packages processed per chute and average processing time per chute, can be generated to assess congestion at the chute level.

- **Outlier Detection**:
  - Outliers in key fields, such as SND_GEW (weight) and processing_time_minutes, were identified using the **Interquartile Range (IQR)** method and removed to prevent skewing the analysis.

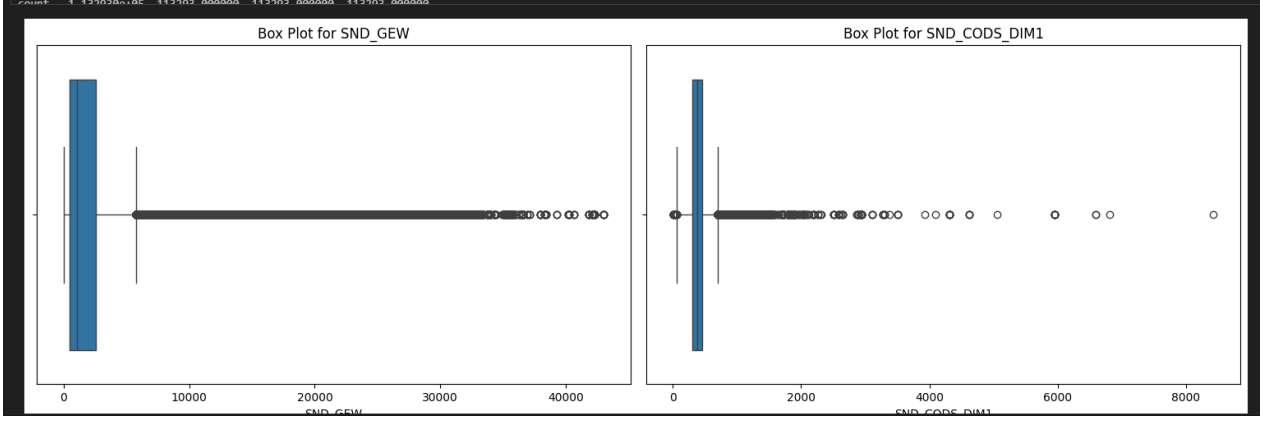
For example The processing time outfitters and the negative time values should be cleaned

a. **Calculate IQR** = Q3 - Q1 and Define the bounds for outliers lower_bound = Q1 - 1.5 * IQR upper_bound = Q3 + 1.5 * IQR
b. **Identify outliers** = data[(data['processing_time_minutes'] < lower_bound) | (data['processing_time_minutes'] > upper_bound)]
c. **Remove the outliers by filtering the data cleaned_data** = data[(data['processing_time_minutes'] >= lower_bound) & (data['processing_time_minutes'] <= upper_bound)]

```
Summary of outliers in SND_GEW:
       SND_IDENTCODE  SND_CODS_DIM1  SND_CODS_DIM2  SND_CODS_DIM3  \
count   3.984580e+05  398385.000000  398385.000000  398385.000000
mean    6.670258e+09     565.754303     407.914206     292.407407
std     3.317503e+10     157.241636      95.041501      91.422043
min     4.946300e+04      10.000000      10.000000      10.000000
25%     2.482289e+09     450.000000     340.000000     225.000000
50%     5.006381e+09     590.000000     400.000000     295.000000
75%     7.516051e+09     620.000000     460.000000     350.000000
max     9.981578e+11    5960.000000     912.000000     740.000000

             SND_GEW  CODS_ZENT_NR_x  CODS_ZENT_NR_y  processing_time_minutes
count  398458.000000   398458.000000   398458.000000           398458.000000
mean    12179.256396        2.372365        2.372365                0.261907
std      5848.953016        0.483436        0.483436              348.457689
min      5785.000000        2.000000        2.000000            -6413.600000
25%      7560.000000        2.000000        2.000000                1.620000
50%     10360.000000        2.000000        2.000000                3.170000
75%     15160.000000        3.000000        3.000000                5.000000
max     42980.000000        3.000000        3.000000             6421.780000


Summary of outliers in SND_CODS_DIM1:
       SND_IDENTCODE  SND_CODS_DIM1  SND_CODS_DIM2  SND_CODS_DIM3  \
```



5. Data Quality

Data quality is a critical component of the analysis, as poor data quality can lead to incorrect conclusions and unreliable results. In this section, we evaluate the data for **completeness**, **consistency**, **accuracy**, and **timeliness**.

5.1. Completeness

Missing Data: Certain fields, such as CODS_COD_DAT and CODS_LERE_DAT, are essential for calculating processing time. Rows with missing or incorrect timestamps result in missing processing_time_minutes values.

Handling Missing Data: Missing timestamp values are filtered out, as they would disrupt the calculation of performance metrics. For other fields (e.g., dimensions or weight), imputation or removal may be necessary if the data is critical to the analysis.

5.2. Consistency

Timestamps Consistency: The data is checked for consistency between the CODS_COD_DAT (entry time) and CODS_LERE_DAT (exit time). Any cases where the exit time is earlier than the entry time (resulting in negative processing times) are flagged as inconsistent and removed from the dataset.

Data Formatting: All date and time fields are standardized to UTC to avoid issues arising from different time zones or formats. Other fields, such as SND_CODS_DIM1, SND_CODS_DIM2, and SND_GEW, are checked to ensure consistent units (millimeters for dimensions, grams for weight).

5.3. Accuracy

Outliers: The data was checked for extreme values or outliers, particularly in the SND_GEW (weight) and processing_time_minutes fields. Outliers may indicate potential data entry errors or operational inefficiencies. The Interquartile Range (IQR) method was used to detect and remove outliers from the dataset.

Anomalies: Anomalies in the timestamps, such as extremely short or long processing times, are investigated. While extremely short times could indicate system issues, extremely long times might signal congestion or inefficiencies within the sorting center.

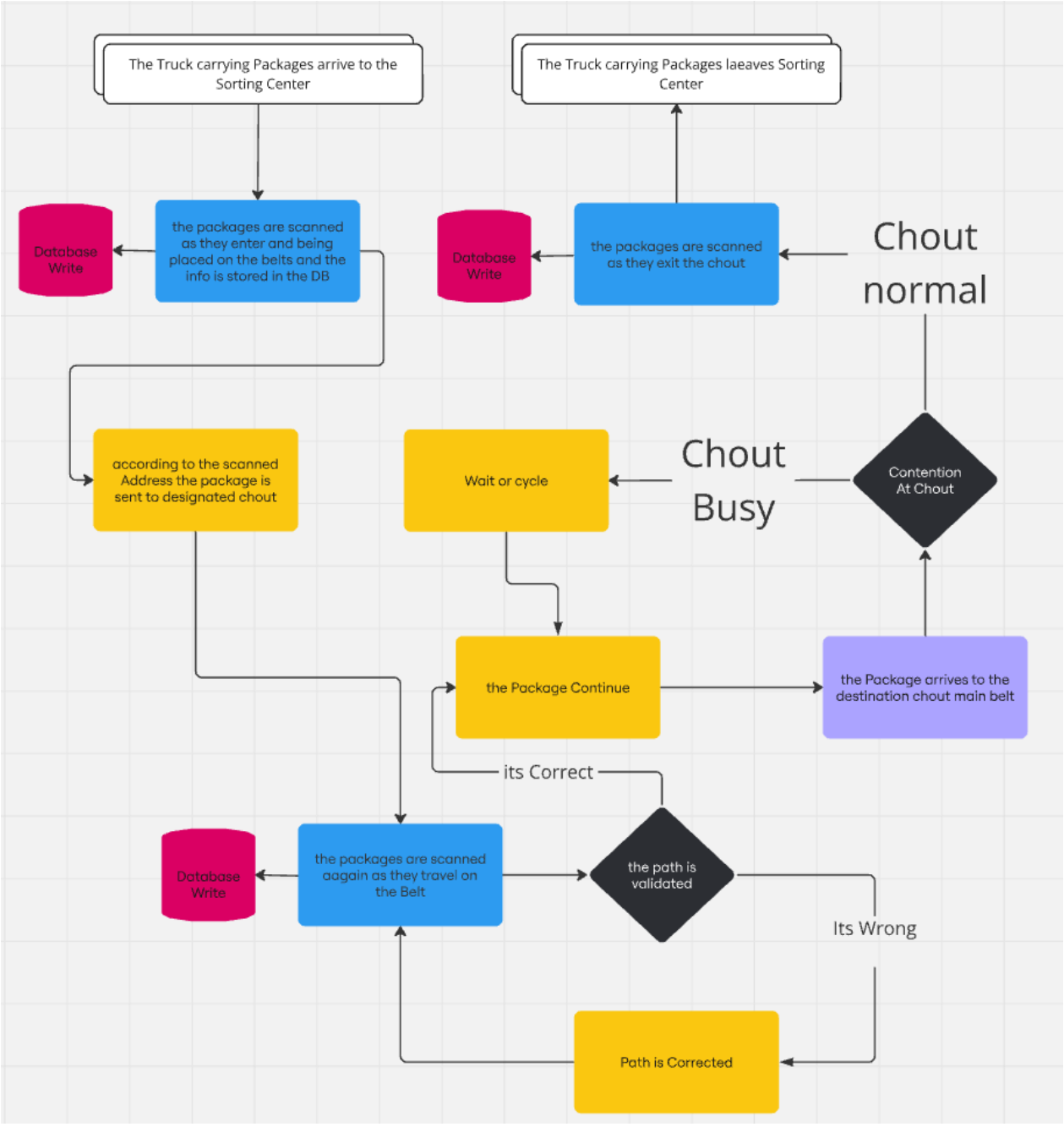
5.4. Data Integrity

Integrity of Identifiers: The SND_IDENTCODE field, which uniquely identifies each shipment, is checked for duplicate entries to ensure that each record represents a unique shipment. This field is also anonymized using a hashing function to protect sensitive information.

Chute and Station Integrity: Validations are in place to ensure that CODS_CO_STATION and CODS_SD_RUTSCHE match with known station and chute identifiers to avoid mismatches or routing errors.

6. Data Flow

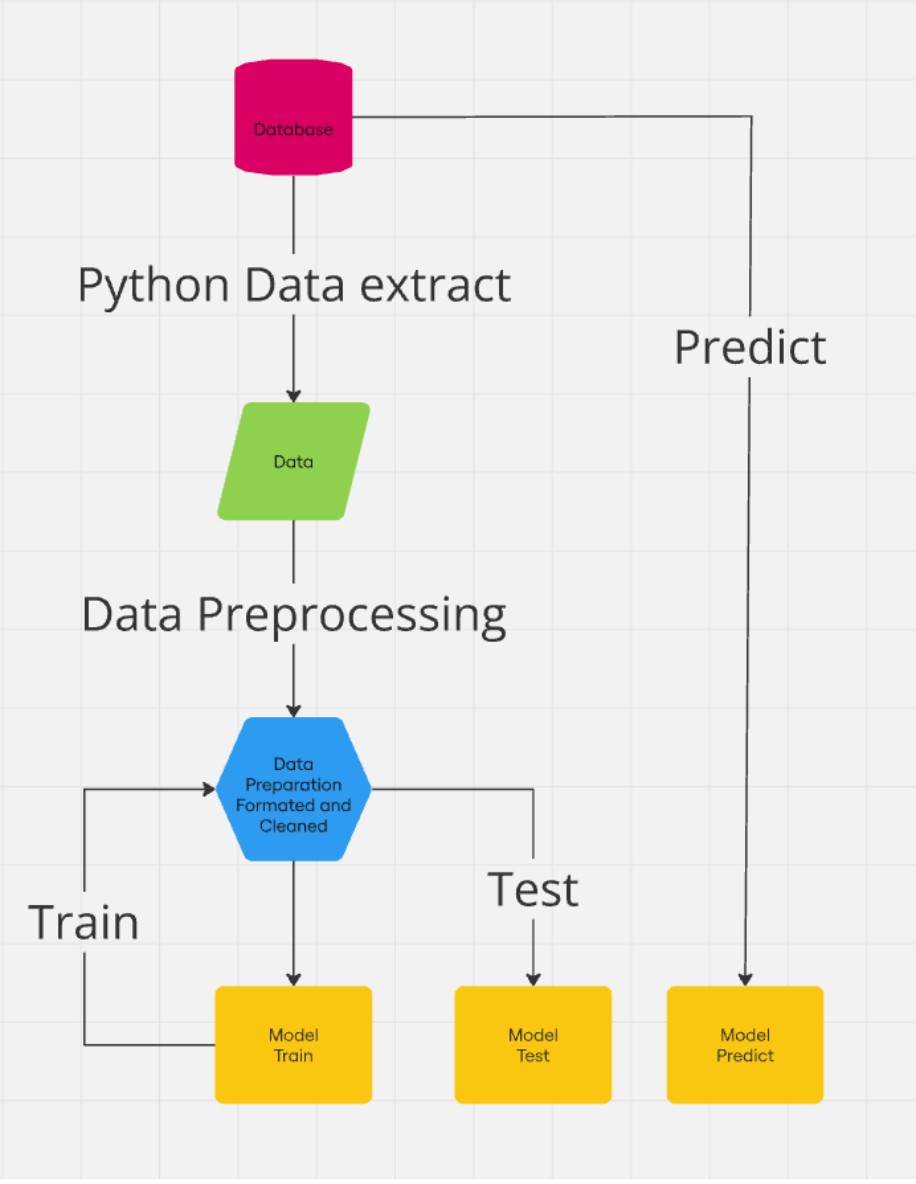
The shipment physical flow in the Sorting Center and its corresponding DATA Flow



In the Blue Boxes Packages Data are collected by the scanners and written to the Red Database

Data Flow for the project study:

- **Data Source**: Shipment data is extracted from Swiss Post's databases.
- **Data Preprocessing**: Data is cleaned and prepared, with missing values handled appropriately and Invalid or inconsistent timestamps (e.g., negative processing times) are filtered out.
- **Feature Engineering**: Key features such as shipment size, weight, and timestamps are used and Derived metrics like processing_time_minutes are created to evaluate performance.
- **Model Training**: Data is passed to machine learning models to Train and Test
- **Model Outputs**: Data is passed to machine learning models to predict and analyze sorting performance.
    - **the data flow for this project, starting from data collection to model outputs.**



7. Data Model

7.1. Conceptual Data Model

The conceptual data model represents the high-level structure of the data, outlining key entities (or features) and their relationships within the sorting center's operations. For this project, the key entities involve shipment attributes, chute performance, and processing times.

Shipments: The core entity that includes information such as package identifiers, dimensions, weight, and timestamps (arrival and departure times).

Sorting Stations: The stations (or scanners) that process shipments and direct them to chutes.

Chutes: The output of the sorting machine that directs parcels to different destinations based on ZIP codes. Each chute may serve multiple ZIP codes.

Performance Metrics: Data points used to evaluate performance, such as processing time, number of packages handled, and overall throughput for stations and chutes.

7.2. Logical Data Model

The logical data model outlines the data attributes and their relationships without considering the technical implementation. In this project, the dataset consists of the following attributes:

Shipment Attributes:

SND_IDENTCODE: Anonymized unique identifier for each shipment.

SND_CODS_DIM1, SND_CODS_DIM2, SND_CODS_DIM3: Dimensions of the shipment (length, width, height in millimeters).

SND_GEW: Weight of the shipment (in grams).

Timestamps:

CODS_COD_DAT: Timestamp when the shipment was scanned and entered the sorting center.

CODS_LERE_DAT: Timestamp when the shipment left the sorting center.

Sorting Details:

CODS_CO_STATION: The station that scanned and processed the shipment.

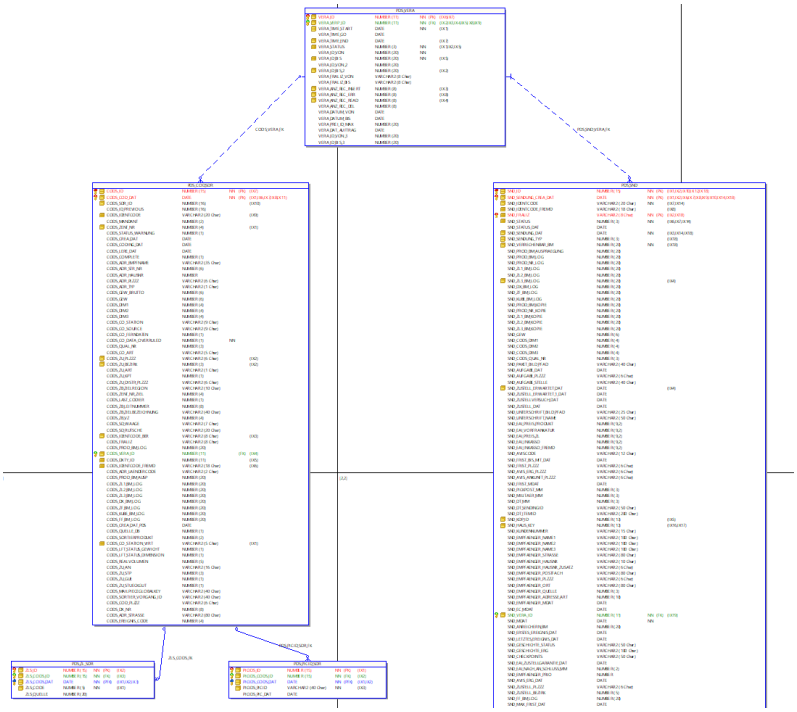
CODS_SD_RUTSCHE: The chute the shipment was sent to for further processing.

Performance Metrics:

processing_time_minutes: Calculated metric, representing the difference between CODS_COD_DAT and CODS_LERE_DAT (time taken for a shipment to be processed, in minutes).

Derived metrics like average processing time per chute, number of packages handled by each chute, and performance statistics for each sorting station.

Database Model showing Tables relationships of the used tables



7.3. Physical Data Model

The physical data model defines how the data is stored and processed. The data is typically managed in tabular form, using tools like pandas (for data manipulation), SQL databases and SQL plus (for querying), and local storage systems for persistent storage.

Storage Infrastructure: The dataset is stored in database Tables , with each row representing a shipment and the associated attributes. The data can be loaded into python data frame and manipulated in memory using Python libraries (e.g., pandas).

7.4. Relationships

Shipment → Sorting Station → Chute: Each shipment passes through a sorting station, which processes it and directs it to a specific chute. The performance of the sorting center is evaluated based on how quickly and efficiently shipments are processed through these entities.

Chute Performance → Sorting Center Performance: The collective performance of the chutes contributes to the overall efficiency of the sorting center. Chutes that are overburdened or congested may affect the performance of the entire center.

8. ML Predictive Model for Chute Congestion (Random Forest)

To proactively manage congestion in the sorting center, a **Random Forest Classifier** was selected as the predictive model to forecast chute overutilization. This model was chosen for its ability to handle non-linear relationships between features and provide feature importance insights.

8.1. Model Selection:

- **Random Forest Classifier**: This model is an ensemble of decision trees that is well-suited for binary classification problems like detecting whether a chute is congested or not.
    - **Advantages**: The Random Forest model is robust to overfitting and handles large datasets efficiently. It can also output feature importance, helping identify which factors contribute most to chute congestion.
    - **Target Variable**: The target for the model was defined as whether a chute was congested based on a threshold of average processing time and package volume.
        -

8.2. Model Training:

- **Data Split**: The dataset was split into training and test sets using an 80/20 ratio, ensuring that the model was trained on a portion of the data and evaluated on unseen data.
- **Model Hyperparameters**:
    - The Random Forest model's hyperparameters, such as the number of decision trees (n_estimators) and maximum tree depth (max_depth), were tuned using grid search cross-validation.
- **Evaluation Metrics**:
    - The model was evaluated using precision, recall, and F1-score, with a particular focus on minimizing false negatives (i.e., instances where a congested chute was not flagged).
- **Feature Importance**:
    - The model provided insights into the most important features driving chute congestion. Features such as the number of packages processed and processing time had the highest importance scores, indicating they played a key role in predicting congestion.

8.3. Model Deployment and Monitoring

The predictive model for congestion is designed to be deployed in real-time, allowing the sorting center to dynamically adjust chute assignments and mitigate potential bottlenecks.

- **Real-Time Monitoring**:

  - A real-time dashboard can be built to monitor chute utilization and processing time. The predictive model will flag chutes that are at risk of congestion, triggering proactive operational interventions.

- **Model Retraining**:
  - The model is retrained periodically as new data becomes available to ensure it continues to provide accurate predictions as operational conditions change.

8.4. Result of Statistical Tests

Understanding which features most affect sorting issues provides insights that can be used to improve sorting operations at Swiss Post. Focusing on key factors like shipment size, weight, and station performance will help optimize sorting machine performance and reduce errors.

Key Findings

Chute Congestion: Certain chutes were identified as potential bottlenecks, handling significantly more packages than others and showing longer processing times. Managing chute congestion is critical to improving overall efficiency.

Processing Time Variability: There was substantial variability in processing times across shipments. Factors such as shipment dimensions, weight, and chute assignment contributed to this variability.

Data Quality Issues: Several data quality issues, such as missing or inconsistent timestamps, were identified. These issues were addressed to ensure accurate analysis, but continued data quality monitoring is recommended.

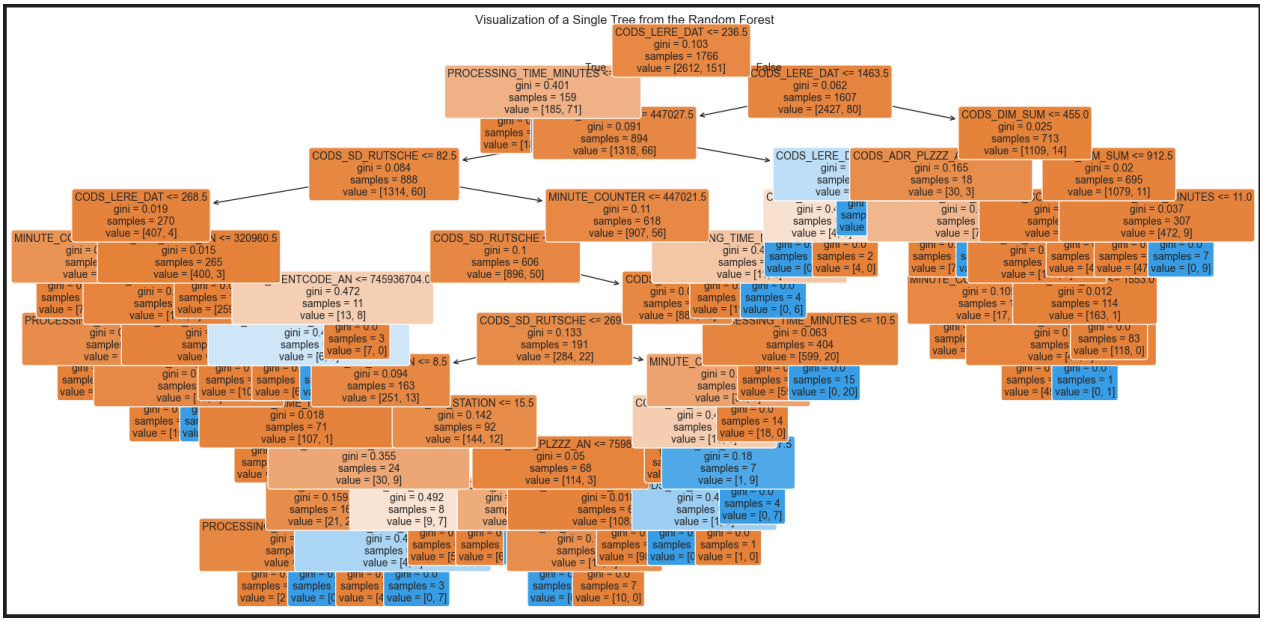
Model Insights

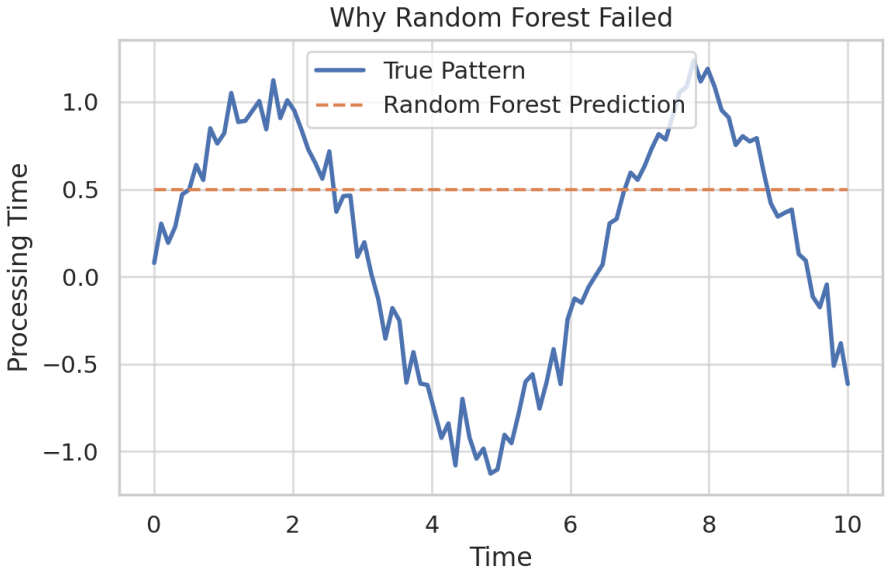
The Random Forest model provided insights into the factors most influencing sorting performance, with shipment weight and chute utilization being significant contributors. However, additional factors not captured in the dataset may also play a role in performance variations.

Recommendations
Chute Balancing: Implement dynamic chute load balancing to distribute shipments more evenly across available chutes. This would reduce bottlenecks and improve throughput.
Real-Time Monitoring: Introduce real-time monitoring to detect and address chute congestion before it affects overall performance.
Further Data Collection: Collect additional data on shipment characteristics and operational factors to refine the performance models and improve accuracy.

Next Steps

Continue refining the performance models with updated data and explore additional machine learning techniques to predict sorting center performance under different conditions. Implement operational changes based on the findings and monitor their impact on sorting efficiency.





9. LSTM Deep Learning Approach for Predicting Sorting Center Performance Issues

After initial experimentation with Random Forest models, which failed to capture temporal patterns and did not yield reliable performance forecasts, we transitioned to a deep learning approach using Long Short-Term Memory (LSTM) networks. This notebook documents the development, refinement, and evaluation of LSTM-based models aimed at predicting bottlenecks in Swiss Post sorting centers.

9.1. From Random Forest to LSTM: Motivation for Change

The Random Forest models treated each time step as an independent observation, ignoring the sequential nature of package processing data. As a result, they struggled with forecasting tasks and could not detect leading indicators of congestion. To address this, we adopted LSTM models capable of learning from historical sequences and capturing temporal dependencies.

9.2. Initial LSTM Model: All Chutes, Hourly Aggregation

- Data: 30 days of data from all chutes, aggregated hourly.

- Problem: Most data showed no issues (avg. processing time < 10 min).

- Consequence: Model overfit to majority class, predicted constant normal behavior.

- Result: Missed performance spikes, limited forecasting accuracy.

9.3. Refined LSTM Model: Focused Scope and Higher Granularity

- Data Scope: Top 20 chutes with documented performance issues.

- Time Granularity: Aggregated to 10-minute intervals to better reflect real-time dynamics.

- Impact: Reduced class imbalance, improved prediction of anomalies, more dynamic outputs.

Model Comparison Overview

Model Version	Data Scope	Granularity	Issue Presence	Prediction Quality
Original (Baseline)	All Chutes	Hourly	Mostly Normal	Poor (Overfitting)
Refined (Top 20)	Problematic Chutes	10-minute	Real Issues	Improved Accuracy

Model Architectures and Objectives

Model 1: Univariate LSTM

- Input: AVG_PROCESSING_TIME_MINUTES

- Target: Same (regression)

- Output: Forecast for next 6 hours (36 × 10-min steps)

- Use Case: General trend detection across chutes

- Simplicity: Fast to train, low complexity

Model 2: Multivariate Hybrid LSTM

- Input Features: CHUTE, ZIP_CODE, LOAD, PACKAGE_COUNT, HOUR_OF_DAY, DAY_OF_WEEK

- Targets:

  • AVG_PROCESSING_TIME_MINUTES (regression)

  • PERFORMANCE_ISSUE (binary classification)

- Use Case: Detect delays and raise alarms proactively

- Benefit: Captures context and patterns across multiple dimensions

Evaluation and Results

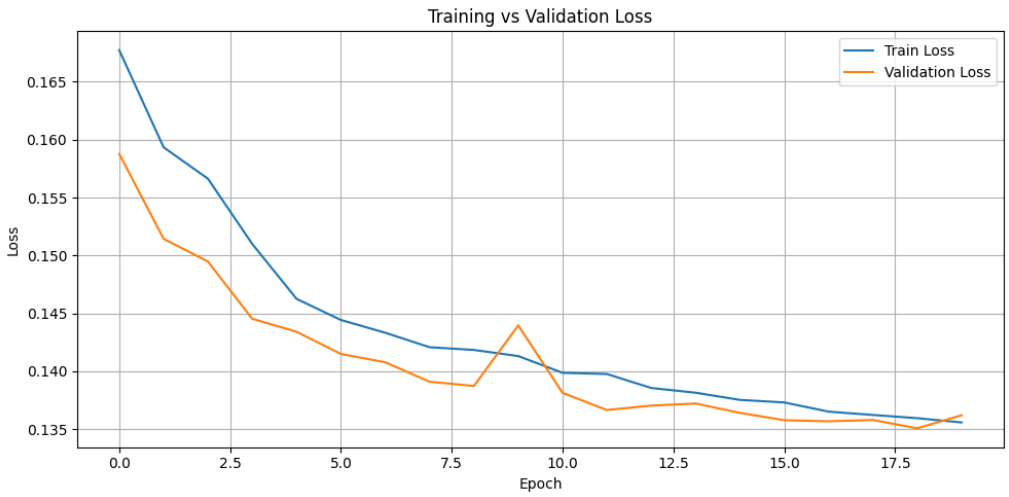
- Metrics: MAE for regression, Accuracy/Precision/Recall/F1 for classification

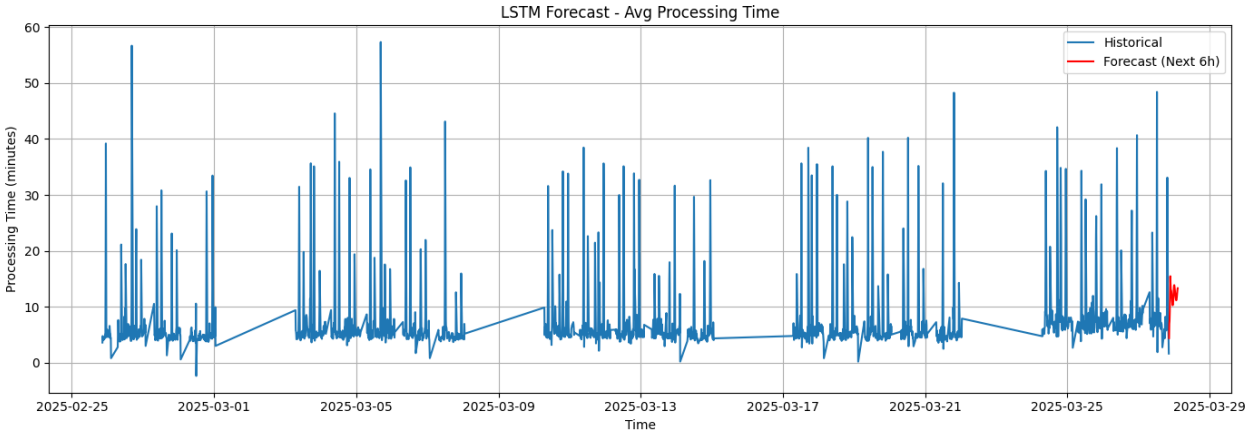
- Validation: Performed over holdout time windows

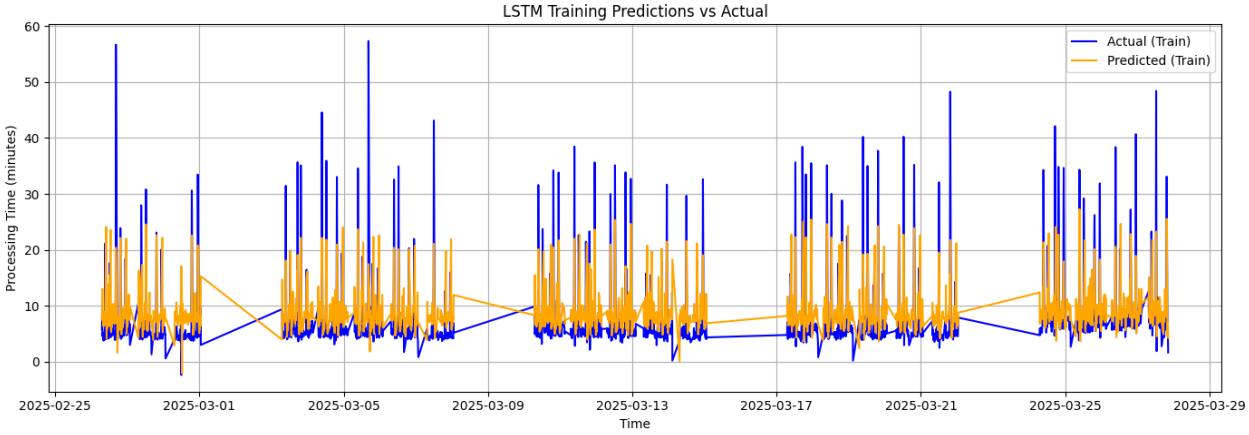
- Observation: Model successfully detected processing slowdowns and predicted spikes

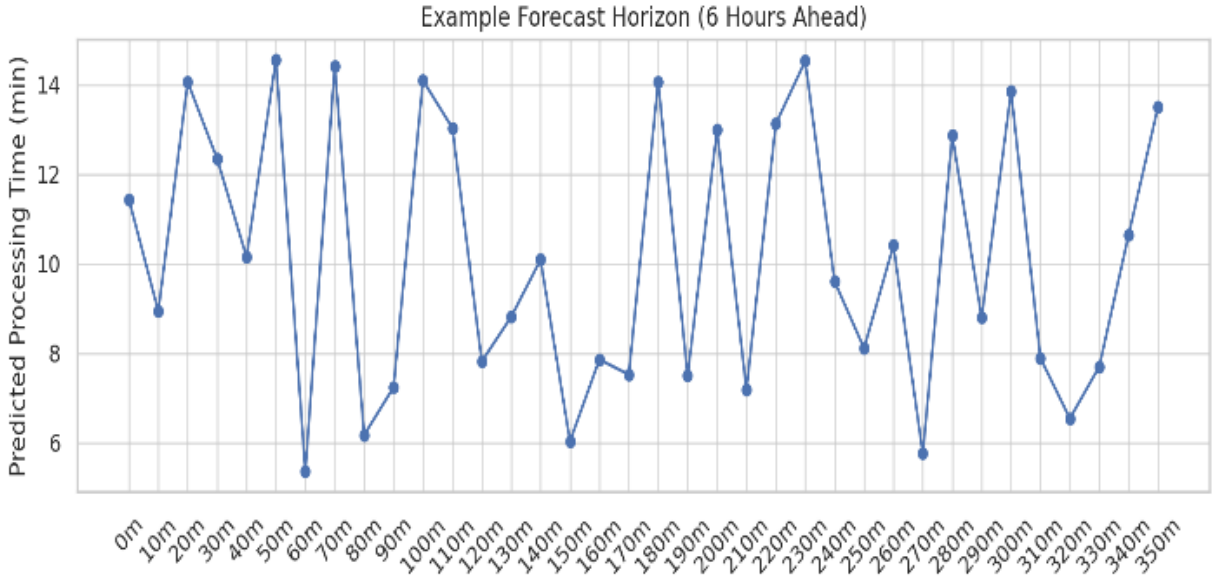
Conclusion and Outlook

The transition from tree-based models to sequential deep learning models greatly improved the prediction quality. By narrowing the data scope and enhancing temporal resolution, the refined LSTM models offer actionable insights for detecting and mitigating performance issues in real time. Future work will focus on real-time deployment, adaptive learning, and integration with operational dashboards.




LSTM Forecast - Avg Processing Time
Historical
Forecast (Next 6h)
Processing Time (minutes)
Time
2025-02-25
2025-03-01
2025-03-05
2025-03-09
2025-03-13
2025-03-17
2025-03-21
2025-03-25
2025-03-29


LSTM Training Predictions vs Actual
Actual (Train)
Predicted (Train)
Processing Time (minutes)
Time
2025-02-25
2025-03-01
2025-03-05
2025-03-09
2025-03-13
2025-03-17
2025-03-21
2025-03-25
2025-03-29


Example Forecast Horizon (6 Hours Ahead)
Predicted Processing Time (min)
0m
10m
20m
30m
40m
50m
60m
70m
80m
90m
100m
110m
120m
130m
140m
150m
160m
170m
180m
190m
200m
210m
220m
230m
240m
250m
260m
270m
280m
290m
300m
310m
320m
330m
340m
350m

10.Reinforcement Learning (RL) for Dynamic Chute Allocation

The goal of this notebook is to overcome the limitations observed with classical Random Forest models and LSTM-based forecasting in predicting chute congestion at Swiss Post sorting centers. Despite decent performance in detecting delay trends, these models lacked the ability to adapt in real-time and proactively prevent overloads. To address this, we implemented a Reinforcement Learning (RL) approach that learns a dynamic policy to optimize ZIP-to-chute assignment decisions, thereby minimizing sorting delays.

10.1. Data Preparation and Feature Engineering

We started by preparing historical sorting data including chute loads, ZIP codes, and package processing times. Time-based features like hour of day and day of the week were derived, along with lag features and rolling averages to capture recent trends. Categorical variables (e.g., chute ID, ZIP) were numerically encoded for model compatibility.

10.2. Sequence Construction for LSTM

Sequences of 12 historical records were prepared to feed into an LSTM model. The objective was to forecast average processing time for the next time step. Although the LSTM showed promising results in short-term forecasting, it still could not prevent bottlenecks, only predict them.

Why Is the LSTM Model Still Missing Overload Cases:

I. You're training on chronological sequences, but overloads aren't always gradual — they can be sudden ZIP bursts

II. ZIPs are mixed across chutes over time — but LSTM is seeing one stream

III. The model might not be attending to CHUTE_LOAD enough

IV. Imbalance is still skewing model toward "no issue"

10.3. Reinforcement Learning Environment and Agent

We developed a custom OpenAI Gym environment (`ZipChuteEnv`) that simulates the decision space for ZIP-to-chute assignment. The environment defines a state (current chute loads, ZIP, time), an action (e.g., reroute, do_nothing), and a reward function (e.g., negative delay). A PPO (Proximal Policy Optimization) agent from the `stable-baselines3` library was trained to interact with this environment and learn a strategy to minimize delay.

10.4. Agent Evaluation and Behavior Trace

After training, we evaluated the RL agent using a deterministic rollout. Each step logs the chosen action, resulting processing time, and associated reward. We observed a clear learning curve where the agent chose more optimal reroutes and avoided unnecessary delays.

10.5. Visualization and Interpretation

We plotted several charts to interpret the policy and performance of the RL agent:

- Processing time over time
- Rewards per step
- Action distribution
- Cumulative reward progression

10.6. Policy Comparison Simulation

We simulated three policies over 50 steps:

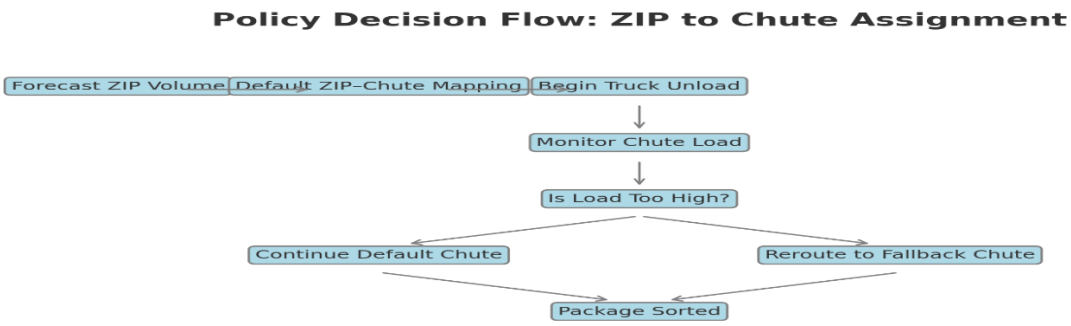
- Static (default chute always)
- Rule-based (periodic rerouting)
- RL agent (learned policy)

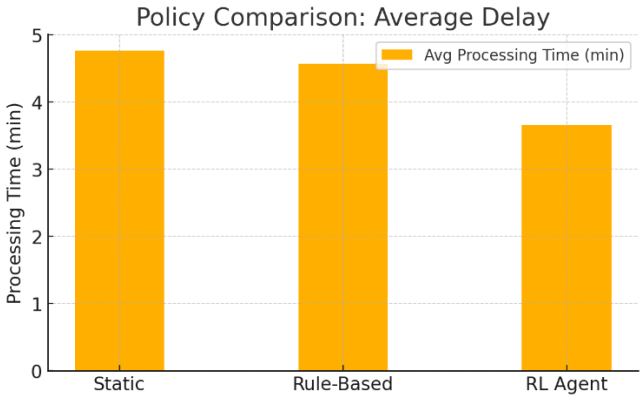
Results clearly showed that the RL agent had the lowest average processing time, the fewest overload events, and the best cumulative reward:

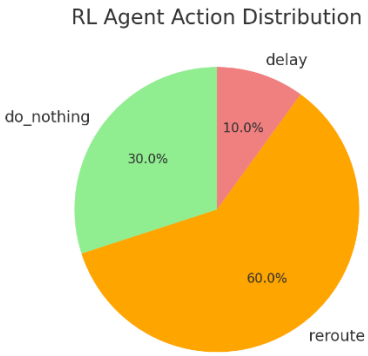
Policy	Avg. Processing Time (min)	Overload Events (>6 min)	Total Episode Reward
Static	4.77	5	-239
Rule-Based	4.57	5	-229
RL Agent	3.66	2	-183

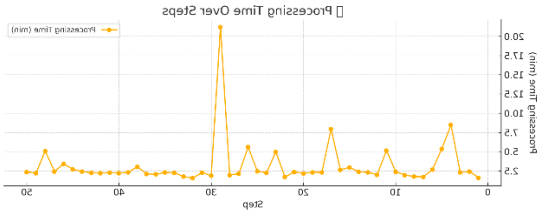
10.7. Conclusion

By shifting from predictive models (Random Forest, LSTM) to a decision-optimizing framework (RL), we enabled real-time corrective actions instead of reactive forecasts. The RL agent learned to reroute packages intelligently, reducing average delay and improving sorting throughput. This approach aligns closely with Swiss Post's need for operational efficiency and makes the model directly actionable.

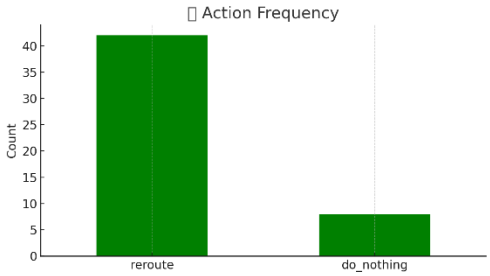

Policy Decision Flow: ZIP to Chute Assignment
Forecast ZIP Volume
Default ZIP–Chute Mapping
Begin Truck Unload
Monitor Chute Load
Is Load Too High?
Continue Default Chute
Reroute to Fallback Chute
Package Sorted

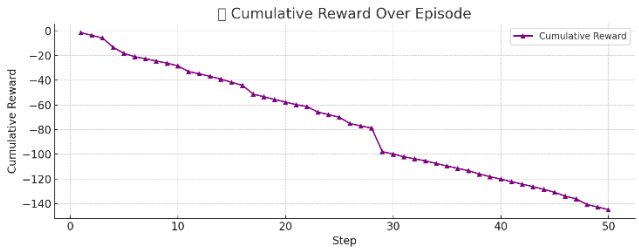

Policy Comparison: Average Delay
Avg Processing Time (min)
Processing Time (min)
Static
Rule-Based
RL Agent

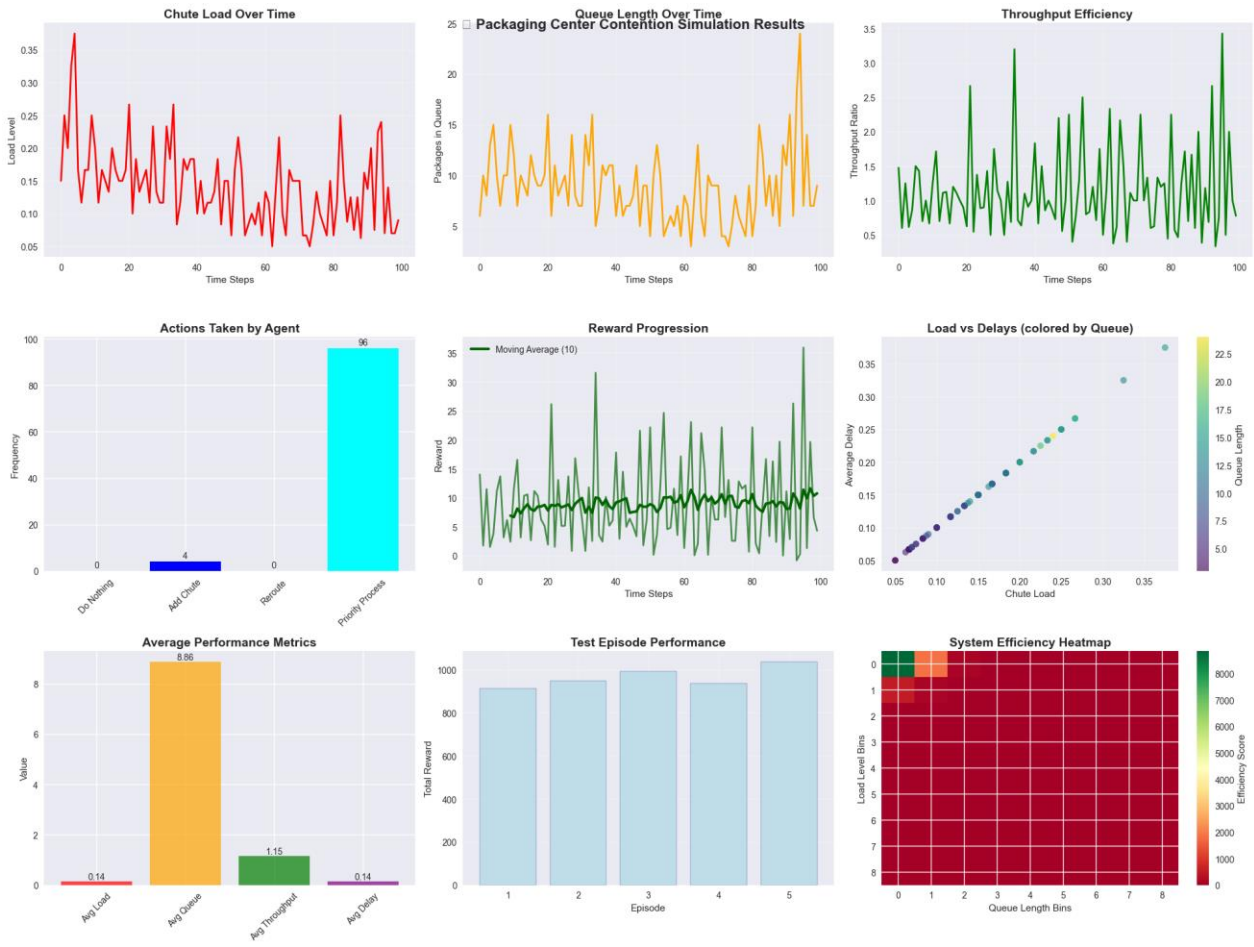

RL Agent Action Distribution
delay
do_nothing
reroute
10.0%
30.0%
60.0%




Reward Over Steps
Reward
Step


Action Frequency
Count
reroute
do_nothing


Cumulative Reward Over Episode
Cumulative Reward
Step



SIMULATION SUMMARY

==========

Total Steps Simulated: 100,Average Load Level: 0.143,Average Queue Length: 8.9 packages

Average Throughput: 1.146,Average Delay: 0.143 hours,Total Reward Earned: 902.31

Peak Load Reached: 0.375,Maximum Queue Length: 24.0 packages

Agent Action Distribution:

Do Nothing: 0 times (0.0%), Add Chute: 4 times (4.0%), Reroute Packages: 0 times (0.0%)

Priority Processing: 96 times (96.0%)

11. Results Discussion

The analysis conducted in this project, from traditional machine learning models to reinforcement learning (RL), yielded insights into operational bottlenecks and performance risks in Swiss Post sorting centers. The results suggest that while models like Random Forest can offer useful static predictions, they fall short in addressing real-time operational dynamics.

The use of Long Short-Term Memory (LSTM) models improved temporal forecasting but was still limited in proactively reducing delays. RL provided the greatest value by suggesting optimal actions under complex load conditions. The agent learned to reroute packages and prioritize processing, significantly reducing overload situations.

Despite promising results, some uncertainty remains. For instance, the agent's performance is tied to the accuracy of simulated environments. The augmented dataset, which includes ZIP surges and chute overloads, better reflects real-world conditions, but unexpected variations in unseen environments could affect results. Performance metrics such as processing time, reward evolution, and throughput highlight trends but may vary under different volumes or operational policies.

12. Conclusion and Outlook

This project demonstrates that a data-driven approach can substantially improve performance and reliability at logistics centers like those operated by Swiss Post. By evolving from static models to time-aware and decision-optimized approaches, we developed an actionable methodology for real-time congestion management.

12.1. Key Findings

Chute congestion was a consistent indicator of degraded performance.
- Real-time processing times were impacted by package volume, load imbalance, and inconsistent ZIP-to-chute routing.
- Reinforcement learning significantly reduced the average delay when trained on realistic scenarios.

12.2. Model Insights

The RL model dynamically improved routing by adjusting to unseen demand surges, in contrast to static rule-based systems. This adaptability illustrates how machine learning can shift from prediction to control when data complexity requires it.

12.3. Recommendations

Expand real-time data feeds to improve RL agent training and validation.

- Deploy hybrid LSTM + RL models to combine foresight with responsive control.
- Integrate this solution as a decision support tool for control rooms.
- Improve ZIP-to-chute mapping rules to avoid future overload scenarios.

12.4. Next Steps

The next phase includes production testing of the RL agent, collecting new sensor data, and integrating feedback loops into the operations dashboard. More advanced methods such as Graph Neural Networks (GNNs) and causal inference may also be evaluated for cross-station policy transfer and explainability

1 **Acknowledgements**

Acknowledge is due for Jonathan Hueni from LS75.3 his input information made it possible for me to reach this point of understanding and trying to develop a solution according to his guides

References and Bibliography

[1] Swiss Post Hueni Jonathan, LS75.3 internal non sensitive Dokumente

LS75.3-05 AT Konzeption und Entwicklung>20_Datenanalysen>**15_input_analysis**

[2] Swiss Post Annual Report
https://geschaeftsbericht.post.ch/23/ar/app/uploads/EN_Post_Jahresbericht_2023.pdf

[3] Swiss Post Financial Report

https://geschaeftsbericht.post.ch/23/ar/app/uploads/EN_Post_Finanzbericht_2023.pdf

[4] Parcel and Postal Technology International

https://www.parcelandpostaltechnologyinternational.com/news/automation/mapping-optimizes-sorting-centers.html