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

Tobias Böni
Wissenschaftlicher Assistent
Berner Fachhochschule
tobias.p.boeni@gmail.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. A single chute can serve several ZIP codes.

The centers 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.

The specific objectives of this project are as follows:

- **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
- **Determine chute congestion impact:** Determine whether chutes are handling disproportionately large volumes of packages can create bottlenecks that reduce overall system efficiency
- **Predict Sorting Performance and Issues**: Develop a predictive model capable of forecasting sorting issues based on historical shipment data.
- **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.

Key components of the infrastructure include:

- 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.
- 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.
- 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.
- 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.
- Sorting centers in Härkingen, Frauenfeld, Daillens and two new centers in Wallisellen and Pratteln 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 these databases, application servers, and a robust network architecture, serve 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.

2.1. 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

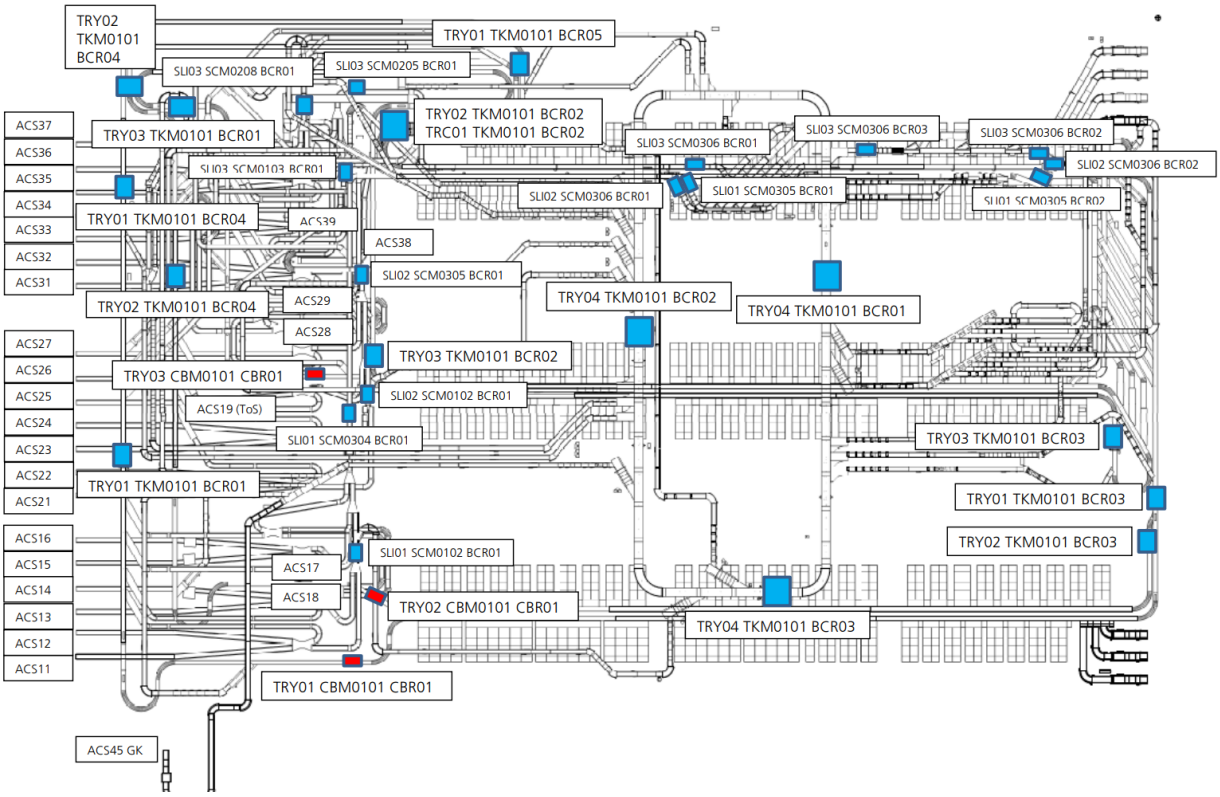
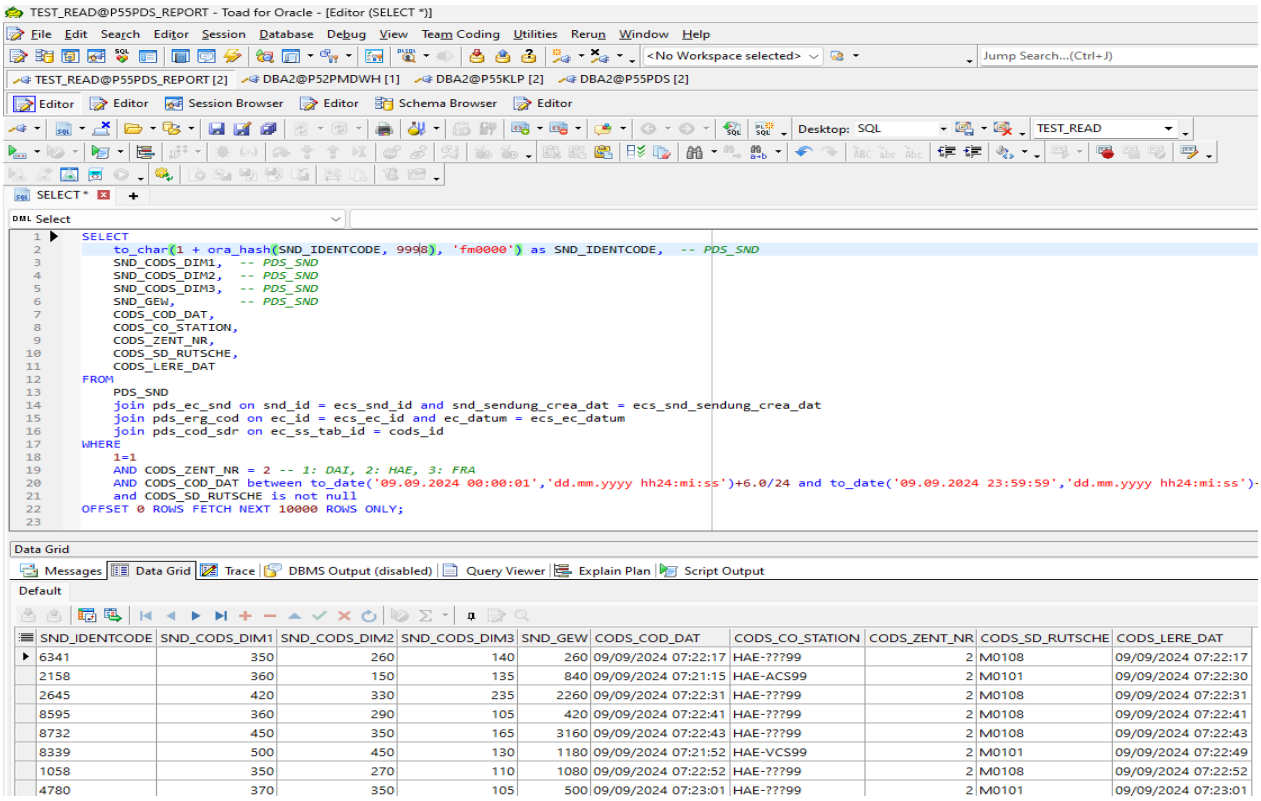


Fig. 1: Schematic of the sorting center in Härkingen in 2022 as an example of the complexity of such centers

3. Data

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

- **Shipment number** (SND_IDENTCODE):
  An anonymized code which is used to track the shipments during their delivery process.
- **Shipment dimensions** (SND_CODS_DIM1, SND_CODS_DIM2, SND_CODS_DIM3):
  Length, width, and height of the package measured in millimeters.
- **Shipment weight** (SND_GEW):
  Weight of the package measured in grams.
- **Scanning timestamps (**CODS_COD_DAT):
  Time and date when the item is first scanned in the sorting center.
- **Scanner station** (CODS_CO_STATION):
  Sorting station identifier.
- **Sorting center number** (CODS_ZENT_NR):
  Indicates which sorting center handled the package.
- **leaving timestamps** (CODS_LERE_DAT):
  Time and date when the item left the sorting center chute.
- **chute station** (CODS_SD_RUTSCHE):
  Which chute is being used for the package sorting according to its destination.



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

Fig. 2: Shows an example of how the data was extracted from the Swiss Post data base using SQL.

Since the data is sensitive, the shipment numbers anonymized to protect the privacy of customers and in turn the data we are using for this project is stored on secure servers of the Swiss Post and access is controlled according to the Swiss Post's privacy and security guidelines.
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. The data used for this project was extracted under security measurements via SQL queries from the Swiss Post servers, preprocessed by these queries and then exported as CSV-files.

4. Statistical Descriptive Analysis

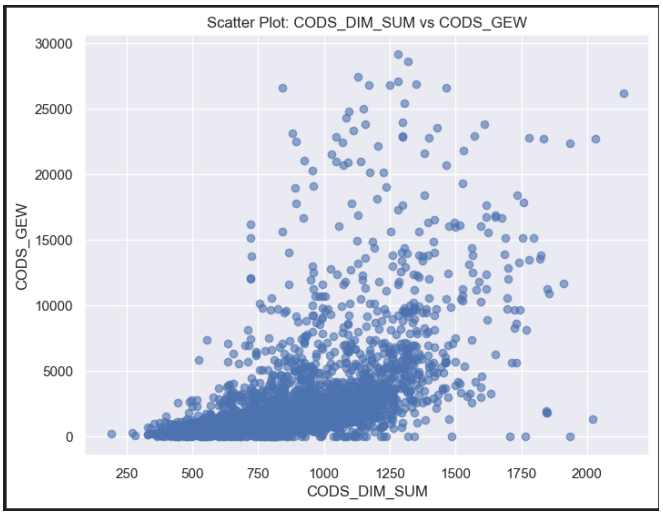
To explore and understand the data more in depth, we started by doing statistical descriptive analysis. This included simple histograms, boxplots and heatmaps. Some notable graphics will be shown further into this chapter. Some notable findings from this analysis include:

- **Chute Utilization**:
  To have a proper measurement value to quantify congestions, we needed to calculate the number of packages processed per chute and the average processing time per chute. Doing this allows us to identify chutes which handle a disproportionally large number of shipments and thus

are more likely to experience congestion which in turn would be reflected by the average processing time of said chute.

- **Performance Metrics**:
  Following the chute utilization metrics we could then identify the performance of different sorting centers and detect centers which experience more congestions. Focusing on the center which averages the most congestions as the impacting factors are likely to more present as well.

- **Data Correlation:**
  The correlation between specific shipments or packages and performance bottlenecks were analyzed to determine if certain supplier lots arriving at the center are contributing to the overutilization of chutes or if certain dimensions impact each other. We believe that identifying these patterns would help us implement proactive redistribution measures to prevent bottlenecks and optimize center efficiency in the future.



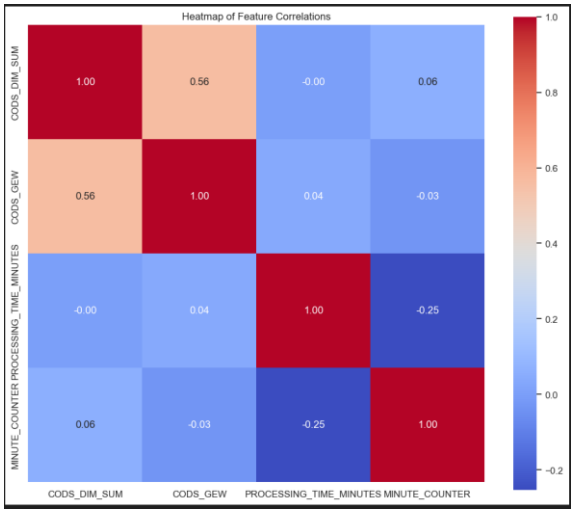
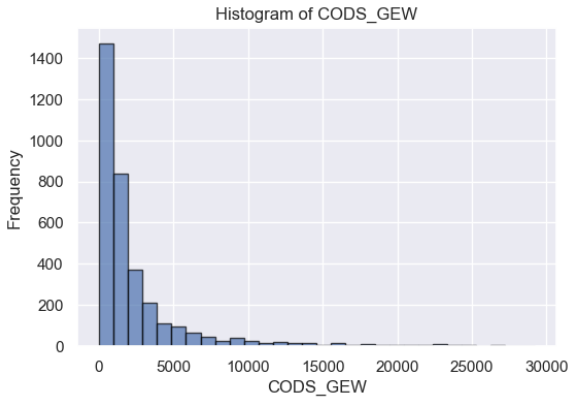


Fig. 3: Scatter plot depicting the distribution of weight and size dimension for packages

Fig. 4: Heatmap showing the correlation of features



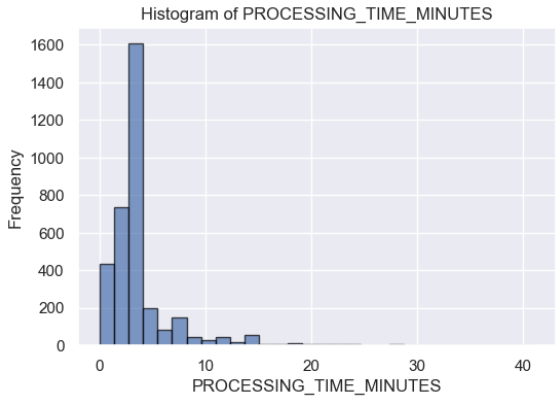


Fig. 5: Distribution of weight among packages

Fig. 6: Distribution of average processing times

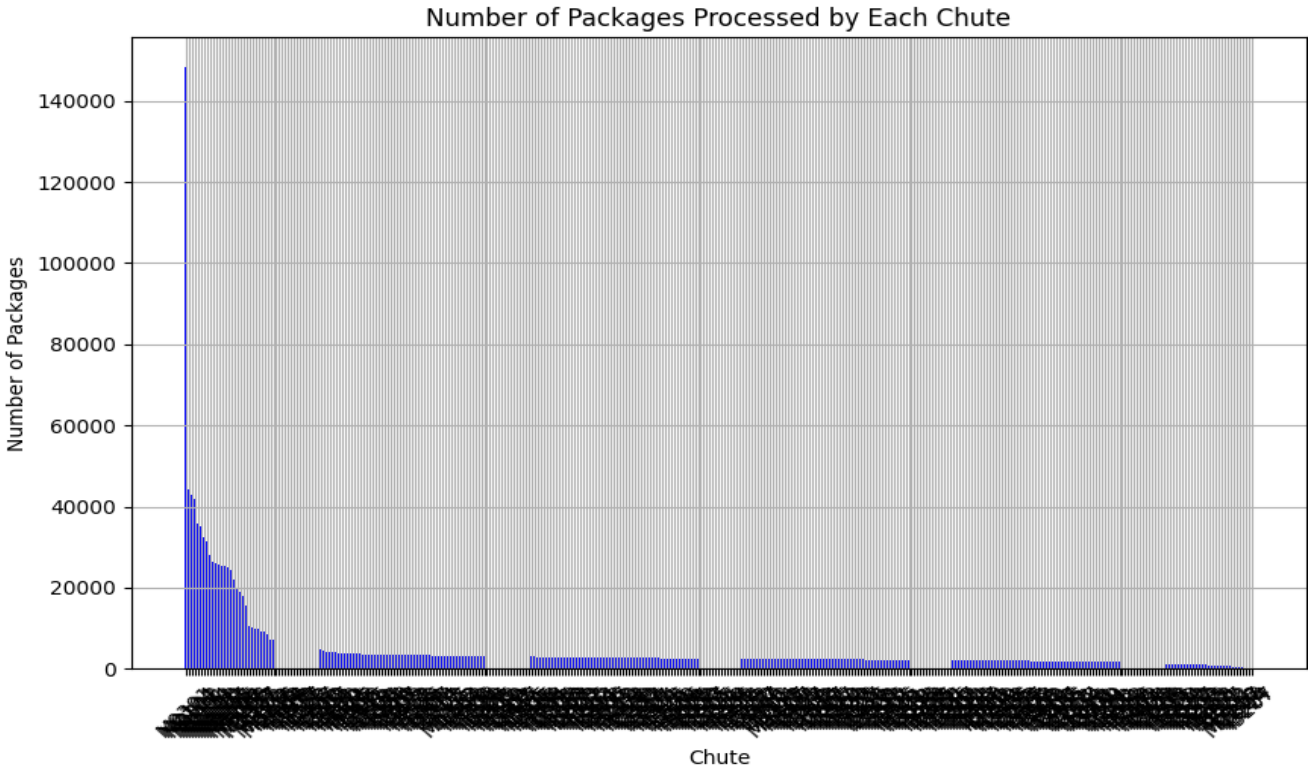


Fig. 7: Shows the number of packages sorted by the individual chutes

4.1. 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 were filtered out and inconsistent data, such as negative processing times, were also removed. As there must have been a measurement error somewhere in the data which we could not fix using the methods we have at our disposal.
- **Feature Creation**:
  To be able to properly quantify the productivity of the sorting centers and the time a package takes to enter and leave a sorting center
- **Outlier Detection**:
  Outliers were identified using the Interquartile Range (IQR) method and removed to prevent skewing the analysis.
- **Integrity of Identifiers**:
  The anonymized serial number of the packages which uniquely identifies each shipment, is checked

for duplicate entries to ensure that each record represents a unique shipment.

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

4.2. Calculating Performance

As previously mentioned, we needed to create at least two new features. To evaluate the performance of the sorting center, we introduced a new column that captures processing time, which is a key performance metric. These metrics we then used to calculate the overall performance of a sorting center.

- **Processing Time:**

  [processing time] = [time of the entering the center]- [time of leaving the center]

- **Center Performance:**

  [center performance] = SUM([processing time] / [package count])

4.3. Data Flow

In following chapter, we try to rudimentarily describe the process of the data collection including the physical flow in the sorting center and when and where measurements are taken. (Fig. 8)

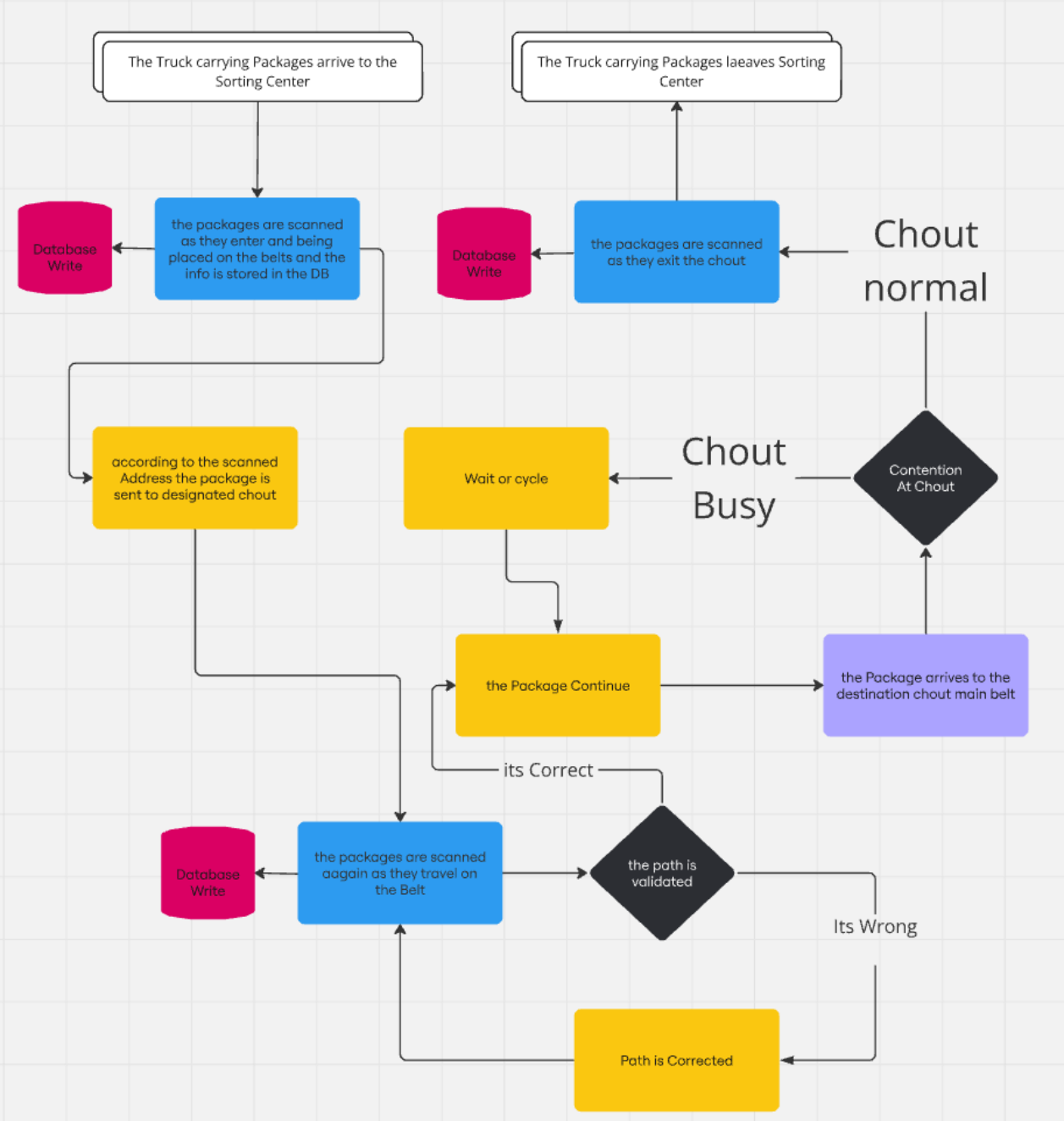


Fig. 8: Schematic process of how a shipment is processed by a sorting center and data is collected

Following the example of the data collection and package processing, the data flow of our future models are described in a similar matter below:

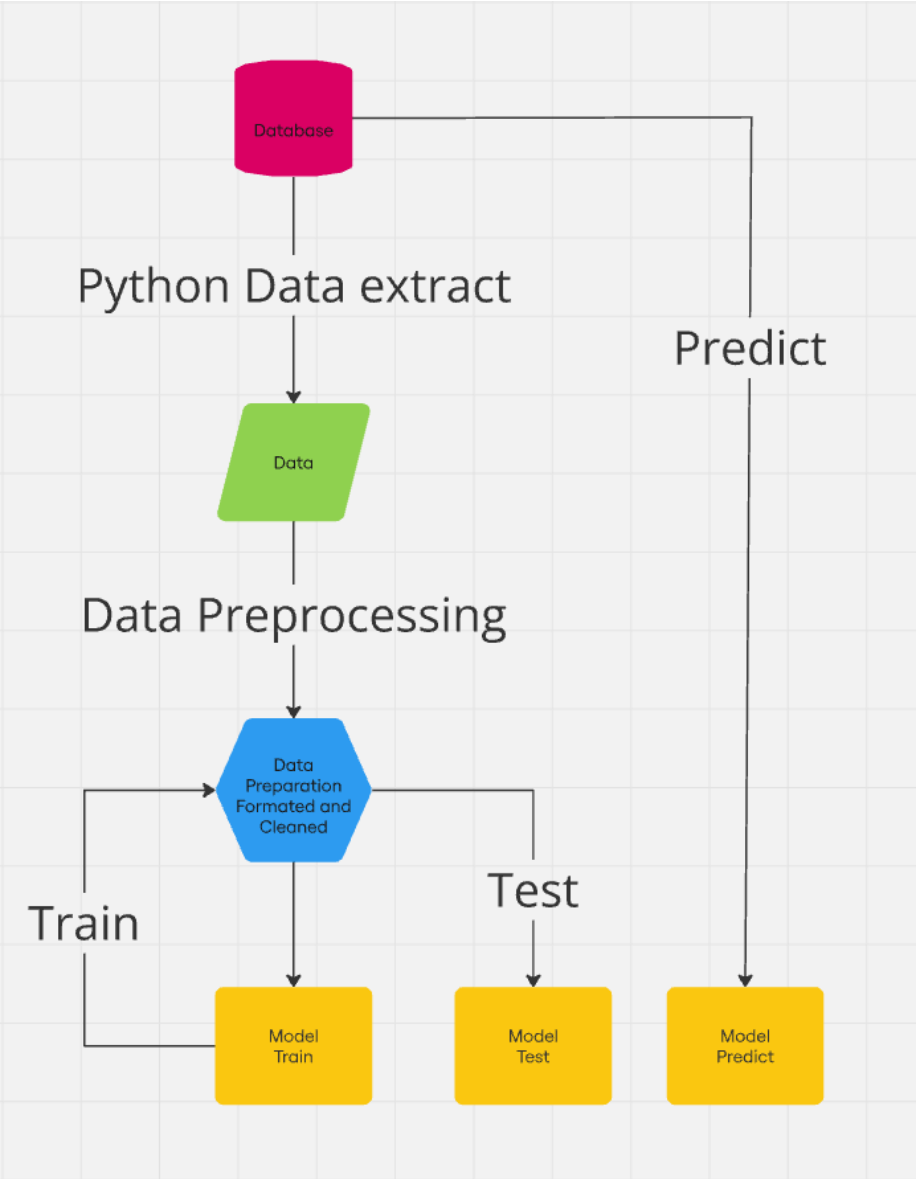


Fig. 9: Schematic process of the data flow in the models used to determine performance issues or doing real-time predictions

Following the schematic in Fig. 9, the data is initially collected from the data source, which in our case is the Swiss Post's databases. This data then is preprocessed by afore mentioned methods and we apply feature engineering to evaluate performance. The models are then trained and tested on the data set which we split 20/80. The model's output is then used to predict and analyze sorting performance and do predictions on live data in real-time.

5. Random Forest Model

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.
Advantages of Random Forest models include its robustness to overfitting and its ability to handle large datasets efficiently. These are both possible pain points of this project that we tried to avoid. Another advantage of Random Forests is that we could define target variables for the model whether a chute is congested or not based on a threshold of average processing time and package volume.

-

5.1. Model Training:

Our 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.
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 was done using precision, recall, and F1-score, with a particular focus on minimizing false negatives (i.e., instances where a congested chute was not flagged).
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 (see 1Random_forest.ipynd).

5.2. 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.

5.3. Results and Findings

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

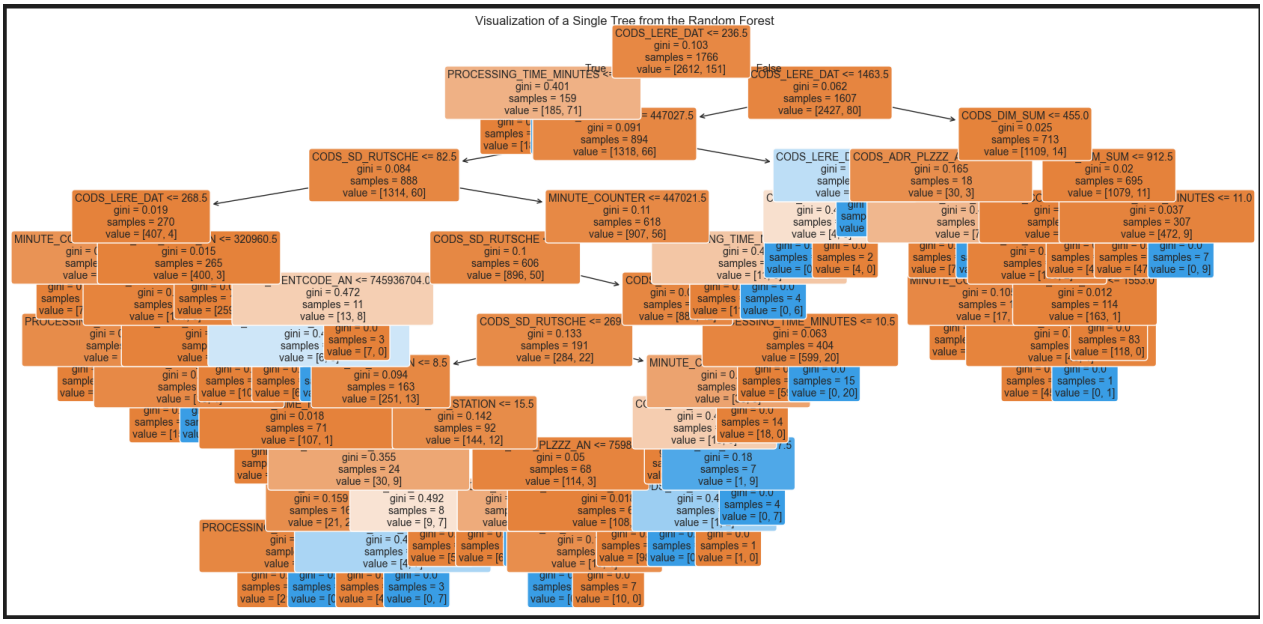


Fig. 10: An attempt to visualize the decision tree made by our Random Forest

While analysing our Random Forest model, which unfortunately did not perform quite as we hoped, we still discovered some 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.

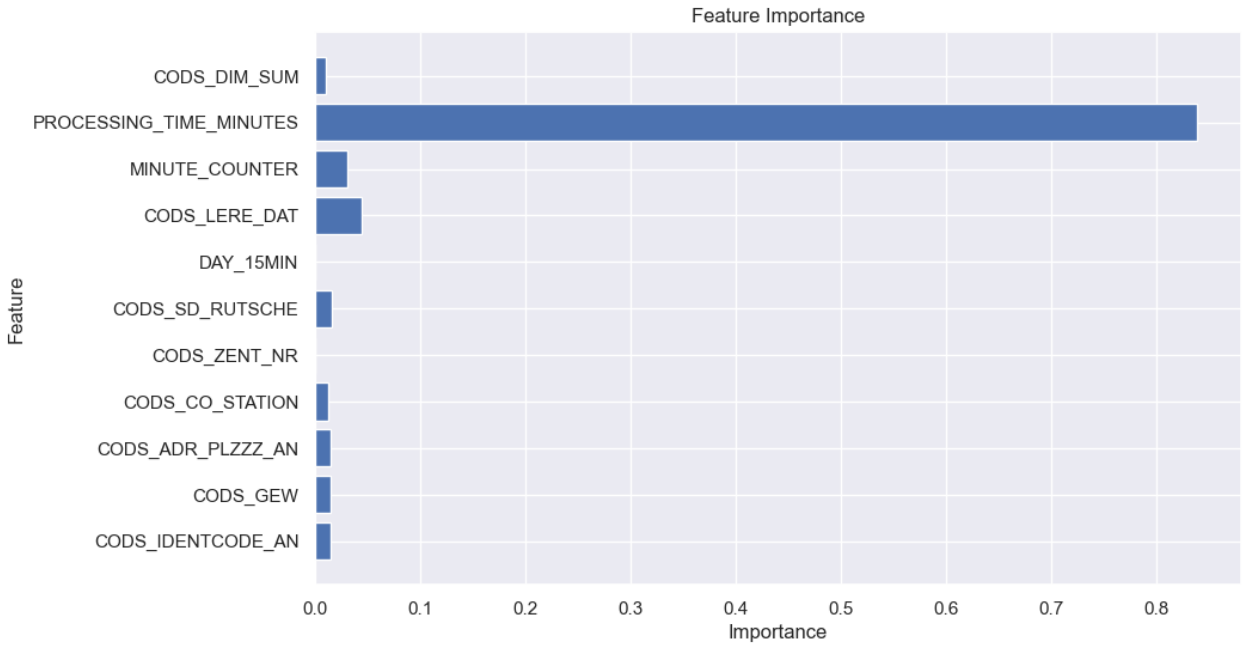


Fig. 11: Feature importance analysis showing that the processing time very heavily impacts the model and is very likely correlated with the target

-

Our Random Forest model further provided insights into the factors most influencing sorting performance with the data that we used (Fig. 11). The processing time proved to be the most important factor which impacts the model performance heavily and suggests that likely need to change our target variable to be the processing time and define congestion according to that value. Additionally, the F1-Scores of 0.973 and Precision of 0.974 for uncongested chutes (class ‘0’) suggest that our Random Forest predicts these very well, however F1 and Precision for congested chutes (class ‘1’) are both 0.0 which in turn tells us the opposite. This discrepancy suggests that the model is possibly biased towards the majority of class ‘0’ as uncongested events significantly outnumber congestion events in the dataset (see Notebook: 1Random_forest.ipynb).

As this model was unable to predict chute congestions for our data set, we tried to embrace our findings and continue with a different approach.

6. LSTM Deep Learning Approach

The Random Forest model treated each time step as an independent observation, ignoring the sequential nature of package processing data. As a result, it struggled with forecasting tasks and could not detect leading indicators of congestion. To address this, we adopted Long Short-Term Memory (LSTM) modelling capable of learning from historical sequences and capturing temporal dependencies. As we are dealing with rather large datasets coming from the sorting centers, LSTM seems like a good fit as the model can store and retrieve information even over long sequences.

6.1. Modelling LSTM

An important change that had to be done from the previous Random Forest model was that we now needed to define congestion via the average processing time which we chose to be 10 minutes. Due to the high availability to data from the sorting centers we decided to take different approaches with LSTM. Another learning from the feature importance we applied was that the dimensions including weight and size of the packages only play a minor role for chute congestion and thus should be dropped from the data set to reduce possible overfitting.
Additionally, we decided to use two different preprocessed data sets for the models. A first data set that included data for 30 days of sorting packages and another one that included only 4 days' worth of sorting data.

LSTM model 1:
Included data of 30 days from all chutes which was aggregated on an hourly basis. With this model we aimed to predict the average processing time of the next 6 hours in a 10-minute frequency.

LSTM model 2:
Was based on LSTM model 1 but added two data engineering steps beforehand as it looked at only the top 20 most congested chutes during those 30 days of data collection and the granularity of time was increased from hourly intervals to 10-minute intervals to better reflect real-time dynamics. For the output was a binary classification chosen which would either be congested or an uncongested chute.

LSTM model 3:
This model used data of 4 days over all chutes. The goal of the model was to predict chutes that were going to experience a congestion and predict their expected average sorting times over the next 6 hours. To increase the number of features for this data set we "enriched" it with simulated time and count lags and rolling means. These additional features are based on the previous counts and times 1 and 3 hours ago (lags) and the averages of both 3 and 6 hours ago (rolling means). This brings the number of features from 5 to 15. The enriched dataset emphasizes chute ID and time-based lags.

6.2. Results and Findings

The transition from the tree-based Random Forest model to the sequential deep learning LSTM model greatly improved the prediction quality. By narrowing the data scope and enhancing temporal resolution

and increasing the number of features, the refined LSTM models offer actionable insights for detecting and mitigating performance issues.

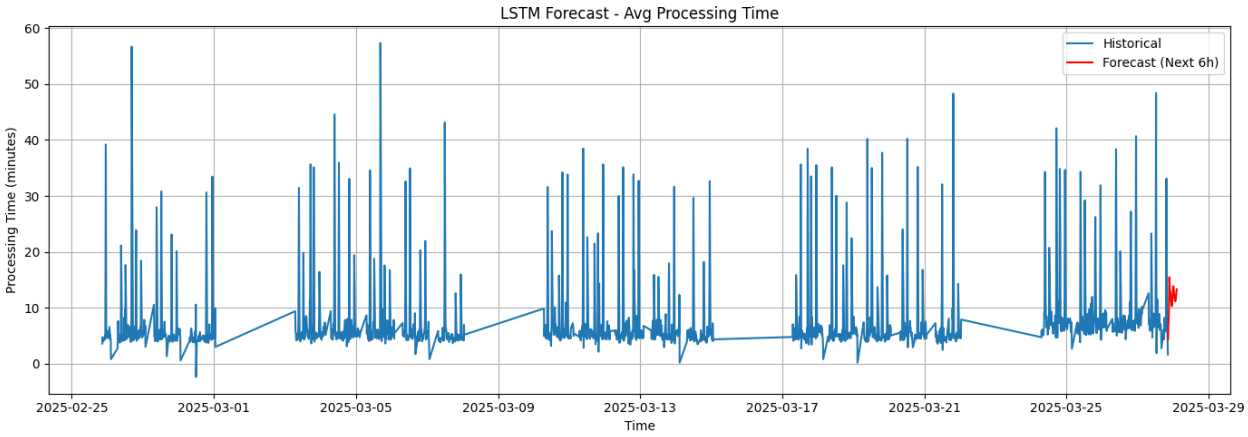


Fig. 12: LSTM model 1 predicting in red the average processing time for the next 6 hours in 10-minute intervals based on the data of 30 days prior.

LSTM model 1 (Fig. 12) predicts an average processing time which seems to vary a bit initially but then does remain around the 10-to-12-minute mark quite consistently, regardless of time window, indicating overfitting to normal class behavior (see Notebook: 2LSTM.ipynb). This would not be what we would expect for a full 6 hours, and neither is it something that was previously observed during the 30 days of data collection, as it would mean that for almost the full duration of the prediction congestions for all chutes would be expected. Due to data imbalance, the model tends to underpredict performance spikes. This highlights the need for better temporal labeling or synthetic event augmentation.

Looking at LSTM model 2, we could see an improvement for peak prediction and less of an 'average' prediction as the previous model did. However, the actual average processing times in this model were incredibly high and these predictions should still be taken with a grain of salt (see Notebook: 2LSTM.ipynb).

For our third model, LSTM model 3, we produced a heatmap (Fig. 13 and Notebook: 2LSTM_enrichedData.ipynb) which shows the hourly predicted congestion times for chutes that experience at least one congestion during the prediction period of 6 hours. The result here seems somewhat reasonable but the enrichment of the data with more features through the lag and rolling means could have influenced the outcome of the prediction in a way we are yet unaware. Also, despite the improved granularity of the model, it exaggerates congestion likelihood, as the average predictions for congestions remain often around 12 minutes, regardless of actual fluctuations.

One major problem while dealing with the LSTM model or modelling in general is that most data show no issues, meaning that the average processing time was under 10 minutes and thus no congestion can be considered. Consequently, the model tends to overfit to majority class and in turn will most often predict normal behavior or simply predicting the average values, missing performance spikes and produce only limited forecasting accuracy.

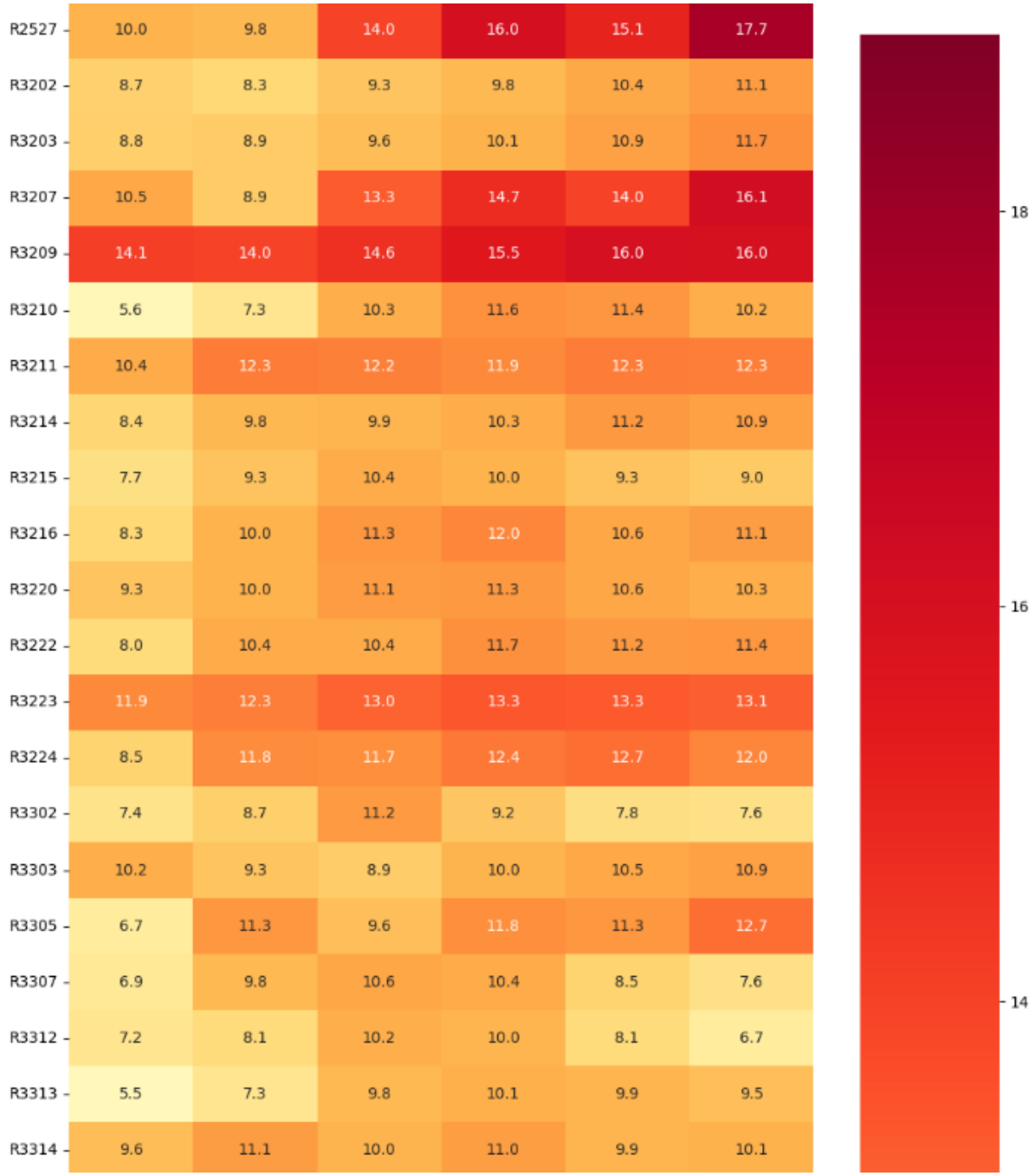


Fig. 13: LSTM model 3, a snippet of the heatmap predicting the average processing times of chutes that experience at least one congestion during the next 6 hours

7. Reinforcement Learning

Despite the improvements the usage of LSTM brought over the initial Random Forest model, we further wanted to enhance the predictive power of our model. For this we decided to build upon the previous LSTM model and use Reinforcement Learning (RL) to reduce the average processing time by letting the RL model take actions in case of an imminent congestion. This approach could enable real time adaptability and as implied proactively prevent chute overloads.

7.1. Data Preparation and Feature Engineering

In a first step we decided to use the historical sorting center data which spanned over four days, based on the LSTM model which we did in the previous chapter. We opted for the enriched data set, which included the additional lag features and rolling average to capture recent trends. As a second step we further artificially enriched the dataset with more congestions to have a dataset that would present a problematic state of the sorting centers performance.

7.2. RL modelling

We developed a custom OpenAI Gym environment which we called 'ZipChuteEnv' which simulates the decision space for ZIP to chute assignment. The environment defines a state (current chute loads, ZIP Code and times), an action (rerouting to a different chute, delaying the sorting process or 'do nothing') and a reward function. 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. We chose to iterate over a maximum of 50 steps which gives the model to choose 50 opportunities to choose the most fitting action. For the dataset with additional congestions, we ran three different complexity models on the data to have an indicator on if the RL agent improves the average processing time or not. These models were a static approach where a package would always be assigned a default chut, a model with periodic rerouting and the model with our RL agent deciding the action.

7.3. Results and Findings

Running our model on the enriched four-day dataset granted us some valuable insights, as the result of 2.88 minutes per step represents a reasonable average sorting time which correlates with the expected values from the dataset which tend to be between 1 to 3 minutes per package if there is no active congestion. Rerouting a package and doing nothing were the only two steps that the RL model chose, where the rerouting action was chosen 42 times out of 50 with an average of 3.41 minutes. The 'do nothing'-action was chosen only 8 times with an average of 2.01 minutes (Fig. 14 and Notebook: 3Reinforcement_Learning.ipynb). This confirms a learned preference for redistributing load rather than passive waiting as seen in a previous RL model (see Notebook: 3Reinforcement_Learning.ipynb).

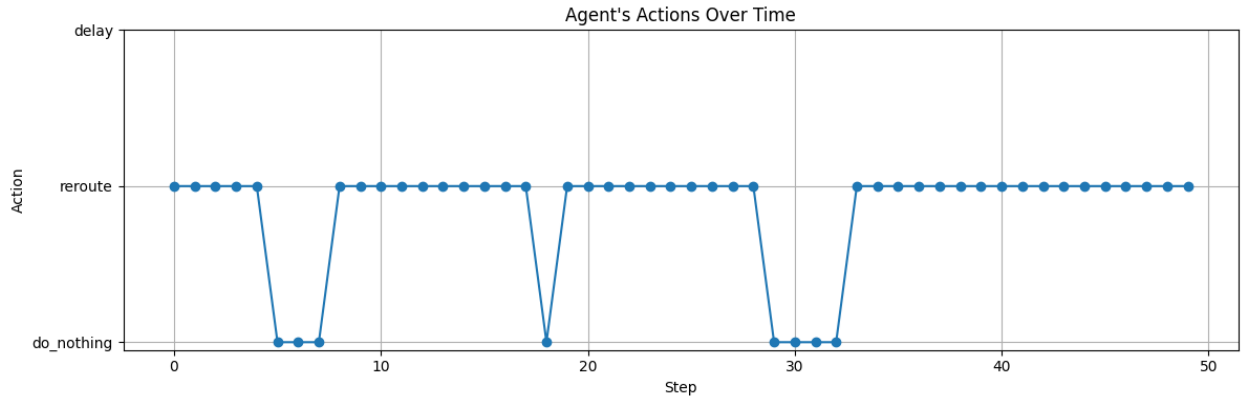


Fig. 14: Actions taken by the RL model for the four-day dataset. Rerouting, despite having a penalty for repeated usage, seems to be the best solution in case of congestions.

The results from the four-day data set enriched with more issues showed us also some interesting results. As shown in the table (Tab. 1) below, our model using the RL agent for decision making improved the average processing time by quite a margin and significantly reduced the number overload events. The RL policy reduced average processing time from 4.77 minutes by the static model, to 4.57 minutes for the rule-based model to finally 3.66 minutes for our RL agent, more than halved overload events from 5 to 2 (Tab. 1). The agent dynamically rerouted on a queue state and ZIP volume, demonstrating policy learning over the 50-step episodes (see Notebook: 3Reinforcement_Learning_With_Issues.ipynb).

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

Tab. 1: Comparison of the three models using the four-day data set with increased frequency in congestions.

By shifting from predictive models (Random Forest and LSTM) to a decision optimizing framework (RL), we enabled real-time corrective actions instead of reactive forecasting.

8. Discussion

The analysis conducted in this project, from traditional machine learning models to Reinforcement Learning, yielded insights into operational bottlenecks and performance risks in Swiss Post sorting centers. As previously shown, the results suggest that while models like Random Forest and even more LSTM can offer useful static predictions. However, they fall short in addressing real-time operational dynamics which is the part which really can improve sorting performance and prevent congestions in real-time. We tried to summarize the results of the most impactful model of each section in the table below (Tab 2.).

Model	Target	Granularity	Avg Proc. Time	F1 (Class 1)	Overloads	Notes
Random Forest	Binary (congestion)	—	—	0.00	—	Strong Class 0 bias
LSTM Model 1	Regression (avg time)	Hourly	~11.5 min	—	—	Flat predictions
LSTM Model 3	Regression	10 min	~12 min	—	—	Better dynamics, still biased
RL Agent	Action Selection	Stepwise	3.66 min	—	2	Best result, real-time capable

Tab. 2: Comparison of our best models with the RL model based on the congestion enriched dataset yielding the best result

Despite promising results, some uncertainty remains. For instance, the agent's performance is tied to the accuracy of simulated environments, which remains rather rudimentary in the scope of this project. 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 may vary under different volumes or operational policies.

9. Conclusion

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 a starting point for an actionable methodology for real-time congestion management. Building on these RL models an accurate real-time predicting method seems reasonable. The next phase could include production testing of the RL agent, collecting new sensor data, and integrating feedback loops into the operations dashboard. However, as previously mentioned some uncertainties remain which need to be taken care before moving to a productive setting.

10. Acknowledgements

Acknowledgements are 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

14. 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

15. Appendix

A

- **1Random_forest.ipynb:**
https://github.com/MoatazM1/CAS_ADS_Post_Final_Moataz_Tobi/blob/main/1Random_forest.ipynb
- **2LSTM.ipynb:**
https://github.com/MoatazM1/CAS_ADS_Post_Final_Moataz_Tobi/blob/main/2LSTM.ipynb
- **2LSTM_enrichedData.ipynb:**
https://github.com/MoatazM1/CAS_ADS_Post_Final_Moataz_Tobi/blob/main/2LSTM_enrichedData.ipynb
- **3Reinforcement_Learning.ipynb:**
https://github.com/MoatazM1/CAS_ADS_Post_Final_Moataz_Tobi/blob/main/3Reinforcement_Learning.ipynb
- **3Reinforcement_Learning_With_Issues.ipynb:**
https://github.com/MoatazM1/CAS_ADS_Post_Final_Moataz_Tobi/blob/main/3Reinforcement_Learning_With_Issues.ipynb
- **data4day_with_issues.csv:**
https://github.com/MoatazM1/CAS_ADS_Post_Final_Moataz_Tobi/blob/main/data4day_with_issues.csv
- **data4day.csv:**
https://github.com/MoatazM1/CAS_ADS_Post_Final_Moataz_Tobi/blob/main/data4day