LUISS

Corso di laurea in Data Science and Management

Cattedra Data Science in Action

Optimizing Voluntary Extra Time Acceptance Prediction:
A Machine Learning Solution for Amazon Logistic Operations

Prof. Alessio Martino

RELATORE

Prof. Blerina Sinaimeri

CORRELATORE

Matr. 766791

CANDIDATO

Anno Accademico 2023/2024

1

LIST OF ABBREVIATIONS

- **AMZL**: Amazon Logistics
- **EU**: European Union
- **UTR**: Under The Roof
- **OTR**: On The Road
- **VET**: Voluntary Extra Time
- **AWS**: Amazon Web Services
- **S3**: Simple Storage Service
- **SVM**: Support Vector Machine
- **MS**: Morning Shift
- **NS**: Night Shift
- **LS:** Late Shift
- **OFD**: operational forecast date

Chapter 1
VET FLEX PROJECT INTRODUCTION

The logistics industry is at the forefront of operational complexity, requiring organizations to continuously innovate to meet customer demands while maintaining efficiency and cost-effectiveness. Amazon Logistics (*AMZL*), operating at a vast scale across the European Union (*EU*)**,** faces the critical challenge of optimizing intra-week workforce planning for its Delivery Stations. This planning, particularly in managing under-the-roof (*UTR*) capacity, directly impacts operational efficiency, customer satisfaction, and overall business performance.

The Voluntary Extra Time Flex project is a response to these challenges, aiming to develop a dynamic framework for managing voluntary extra time (*VET*) shifts. VET plays a pivotal role in filling UTR capacity gaps, especially during periods of fluctuating demand. However, accurately predicting VET acceptance and integrating this into shift planning has proven to be a significant obstacle due to the inherent unpredictability of workforce availability. A critical foundation for this project was the development of an improved forecasting system to predict daily workforce requirements with high precision. The forecasting system was built on a hybrid approach, employing time series models like Prophet to accurately predict daily volume, while leveraging machine learning techniques such as XGBoost for classification tasks related to workforce availability. This combination significantly enhanced prediction accuracy by addressing both operational trends and workforce dynamics. This method accounted for non-linear trends, seasonality, and operational factors such as oversize package share and capacity constraints. The improved forecasts were instrumental in aligning workforce plans with

actual operational demands, minimizing inefficiencies and setting the stage for VET optimization.

This thesis presents a solution to this problem through the development of a dynamic VET acceptance model. By leveraging machine learning techniques, the model provides accurate predictions of the Flex VET, allowing for real-time adjustments to workforce plans. These adjustments ensure that capacity aligns more closely with actual operational needs, minimizing inefficiencies and lost volumes.
The project also seeks to standardize and refine UTR capacity calculations, incorporating key factors such as scheduled headcount, expected absences, productivity rates, and opportunities for VET or voluntary time off (*VTO*). This standardization aims to establish consistency across AMZL's capping platforms, laying a robust foundation for decision-making.

In addressing these challenges, this work employs a combination of data manipulation, preprocessing, and machine learning techniques. The integration of these techniques highlights the importance of a data-driven approach in solving real-world operational problems. The financial impact of this project is hypothesized to be substantial. By increasing the average VET from its current levels and reducing inefficiencies, the model offers potential cost savings of approximately 23% for the UK and MEU regions in 2024. Through this thesis, we document the development and evaluation of the VET model, exploring its potential to revolutionize intra-week shift planning in the logistics sector. By combining predictive modeling with data-driven optimization, this work contributes to the broader goal of operational excellence in workforce management.

Chapter 2
VET MODEL AND AMAZON SUPPLY CHAIN

2.1 **Introduction to the VET Model**

The overall goal of developing the VET Flex forecasting model is to optimize the management of intra-week shift planning for Delivery Stations in the EU, particularly focusing on maximizing UTR capacity while minimizing volume lost due to intra-week UTR capacity gaps. This aligns with the broader business objectives of maximizing AMZL attainment while ensuring operational efficiency and cost-effectiveness.

2.2 **AWS Tools Used in the Project**

During the development of the project, Machine Learning techniques were employed by leveraging the services offered by the Amazon Web Services (AWS) platform. Thanks to AWS, all stages of the project could be executed on a single server, utilizing the connections and tools provided by the platform. Specifically, the following tools were used:

- **Amazon Simple Storage Service (Amazon S3):** an object storage service that provides high scalability, data availability, security, and performance. Amazon S3 was used to capture data from numerous Delivery Stations across Europe and organize them into a single Data Frame. This phase represents the Data Collection process.
- **Amazon Redshift:** a fully managed, fast, cloud-based data warehouse that facilitates the analysis of structured and semi-structured data using SQL. Amazon Redshift enabled data manipulation and preprocessing, as well as the creation of separate databases for each Delivery Station, country, and region. These databases were later used for training Machine Learning models.

- **Amazon Sage Maker AI***:* a fully managed service that provides a wide range of tools for Machine Learning. Amazon SageMaker was used to create a virtual environment with Jupyter Notebook, where various files for the project's analysis were developed.
- **Amazon Forecast:** Amazon Forecast is a fully managed machine learning service provided by AWS that enables businesses to generate highly accurate forecasts using their data. It allows users to predict metrics such as demand, sales, revenue, resource requirements, or inventory needs based on historical time-series data. Amazon Forecast automates much of the machine learning process, making it accessible to users without deep expertise in data science or machine learning.

These tools, seamlessly integrated within the AWS ecosystem, facilitated the optimization of data management and analytical processes, ensuring both consistency and scalability across the pipeline. As part of the Machine Learning workflow, the initial preprocessing steps included feature scaling and the encoding of categorical variables, effectively preparing the dataset for subsequent analysis.

A range of models were explored, including Gradient Boosting, Random Forest, and Support Vector Machines (SVM), with hyperparameter tuning applied to fine-tune model performance and enhanced predictive accuracy. Evaluation metrics such as Accuracy, AUC, MAE, and RMSE were used to guide the model selection process, ensuring a comprehensive assessment of their effectiveness.

2.3 **Challenges Addressed**

Amazon supply chains are like living organisms—constantly adapting, growing, and evolving to meet the demands of customers. For this project, the goal was clear: to make the process of managing workforce ability smarter and more efficient, especially in the Last Mile, where every package and every shift counts. One of the biggest questions we had to answer was: How can we define the willingness of people to say "yes" to volunteer

for extra work? Predicting this “acceptance rate” was not as easy as it sounds. People’s choices are influenced by all sorts of factors, from timing to location to who is asking. To achieve this, we had to tackle many challenges, starting with data cleaning and the entire pre-processing phase, all the way to project optimization, result interpretation, and the creation of a user interface. This interface was designed to enable teams working in Amazon's warehouses—who may not have coding skills—to use the tools effectively. We also had to ensure that the interface provided actionable insights in real time, allowing warehouse teams to make informed decisions quickly and efficiently. This meant integrating key performance indicators, predictive analytics, and intuitive visualizations, all while maintaining a seamless user experience. By bridging the gap between sophisticated data models and practical, day-to-day operations, we aimed to create a tool that not only solved immediate challenges but also laid the foundation for scalable solutions in the future. This required us to develop a user-friendly interface that was both easy to use and packed with valuable information.

2.4 **Amazon Supply Chain Overview**

To fully understand how the VET system works and why it is so valuable within Amazon Logistics, it is essential to explore a few key concepts that will allow us to approach the subject in a clear and comprehensible manner. In the following, we provide a theoretical introduction to the project, delving into critical ideas that are fundamental to the operations of Amazon Logistics today.

> «Amazon’s supply chain is a sophisticated, interconnected system designed to ensure seamless and efficient delivery of goods from suppliers to customers. It operates across three key stages: First Mile, Middle Mile, and Last Mile. Each stage plays a distinct yet complementary role in the end-of-the-end process. »[1]. (*Amazon*, 2024)

The *First Mile* marks the very beginning of the supply chain, where products are transported from suppliers or manufacturers to Amazon’s Fulfillment Centers. This stage focuses on establishing a steady flow of inventory into Amazon's network, ensuring that goods are prepared, shipped, and received in alignment with demand forecasts and operational standards. It forms the foundation for the rest of the supply chain.

[1] Amazon Internal Wiki, Ato2023.

The Middle Mile involves the strategic movement of inventory between Amazon's internal facilities, such as fulfillment centers, sortation hubs, and delivery stations. This phase is characterized by bulk transportation over longer distances, using Amazon's fleet, third-party carriers, and air freight services. The aim here is to position products optimally across the network, balancing speed, cost, and proximity to customer delivery zones. *The Last Mile* represents the final stage, where goods are delivered directly to customers. This highly customer-facing phase relies on Amazon's logistics network, including its delivery fleet, third-party couriers, and independent drivers, to ensure orders reach their destinations quickly and reliably. This integration enables Amazon to consistently meet customer expectations for fast and dependable delivery, setting a benchmark in modern supply chain management.

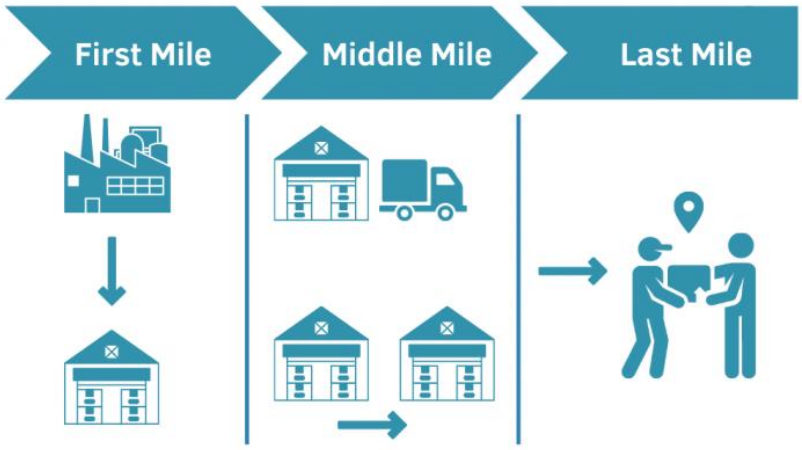

Figure 1: Amazon supply chain easy representation

2.4.1 **Last Mile and Delivery Station**

As anticipated, the Last Mile refers to the final leg of the delivery process, where packages are transported from a distribution hub to their destination, typically the

customer's doorstep. This phase is often considered the most challenging and expensive part of the shipping process due to its complexity and the need for individual deliveries to many locations. The *Delivery Station* is a facility that serves as the last stop for packages before they enter the Last Mile phase. These stations receive shipments from larger fulfillment centers or sortation centers and prepare them for final delivery. When at a Delivery Station, packages are sorted based on delivery routes and loaded onto vehicles for distribution. The process at a Delivery Station typically involves receiving inventory from larger warehouses, scanning and sorting packages, organizing them by delivery zones, and then dispatching them to delivery drivers or couriers. These stations are strategically found in urban and suburban areas to minimize travel time and increase delivery efficiency.

Delivery Stations play a pivotal role in streamlining the Last Mile process by combining shipments, optimizing routes, and ensuring that packages are organized for swift and accurate delivery. They often operate around the clock, with early morning shifts preparing deliveries for same-day or next-day service. In recent years, the importance of Last Mile delivery and Delivery Stations has grown significantly with the rise in e-commerce and customer expectations for faster shipping times. Companies are continuously innovating in this space, exploring modern technologies like route optimization software, automated sorting systems, and even drone deliveries to improve efficiency and reduce costs in this critical part of the supply chain.

2.4.2 **The Last Mile 24 Hour Cycle – Understanding shifts in Last Mile Logistics Operations**

In the Last Mile logistics sector, there are three main shifts that operate around the clock to ensure efficient package delivery. These shifts are often referred to as the Morning shift (DS), Late Shift (LS) and Night shift (NS).

The Day shift, which usually starts early in the morning, around 6 AM or 7 AM, and continues until mid-afternoon, is primarily focused on the actual delivery of packages to customers. This shift is when most of the visible Last Mile operations occur. Delivery drivers begin their day by loading their vehicles with packages that were sorted and prepared during the night. They then embark on their routes, making deliveries throughout residential and commercial areas. The Day shift also handles customer service issues that may arise during deliveries, such as incorrect addresses or package reception problems.

The Twilight shift or Late Shift typically begins in the late afternoon, around 2 PM or 3 PM, and continues into the evening hours. This shift serves as a bridge between the day's deliveries and the night's sorting operations. Twilight shift workers often handle packages that arrive too late for the Day shift to deliver. They may make some deliveries, especially to businesses with extended hours, and begin the process of sorting and organizing packages that have arrived from regional distribution centers. The Twilight shift also often deals with returns and redirected packages, preparing them for processing or re-delivery.

The Night shift, which usually starts in the late evening, around 10 PM or 11 PM, and continues through the early morning hours, is crucial for preparing the next day's deliveries. This shift is primarily focused on receiving, sorting, and organizing packages that have arrived from larger fulfillment centers or sortation facilities. Night shift workers use advanced sorting systems to categorize packages by delivery routes, ensuring that everything is in place for the Day shift drivers to load their vehicles efficiently. They also handle any overnight shipping arrivals and prepare express packages for early morning delivery.

Each of these shifts plays a vital role in the 24-hour cycle of Last Mile operations. The seamless transition between shifts is critical for maintaining the flow of packages and meeting delivery deadlines. While the Day shift is the most visible to customers, the work done during the Twilight and Night shifts is equally important in ensuring prompt and

correct deliveries. The specific timing and duties of each shift can vary based on factors such as location, volume of packages, and company policies. For instance, in urban areas with high package volumes, there might be more overlapping between shifts or even additional micro-shifts to handle peak delivery times.

Moreover, the advent of same-day and next-day delivery services has led to increased flexibility in shift structures. Some Last Mile operations now include rapid-response teams that work across traditional shift boundaries to handle urgent deliveries.

Technology plays a crucial role in coordinating these shifts. Advanced logistics software helps in planning routes, tracking packages, and managing workloads across all shifts. This ensures that each shift has the right number of staff and resources to handle the expected volume of packages. It is also worth noting that the shift structure in Last Mile operations often requires a high degree of adaptability. During peak seasons[2], such as holidays, shifts may be extended, or additional temporary shifts might be added to cope with increased package volumes.

In conclusion, the multi-shift structure of Last Mile operations enables a continuous flow of package processing and delivery, ensuring that the ever-increasing demands of e-commerce and customer expectations for swift delivery are met efficiently and effectively.

2.5 **A brief introduction to workforce planning optimization and volume forecasting**

The use of advanced optimization models, such as the VET Flex, arises from the need to address increasingly complex challenges in shift planning within Amazon's supply chain. Specifically, the Last Mile requires precise adaptability to respond to fluctuations in operational volume while ensuring efficiency and sustainability. Before implementing the VET FLEX model, a critical step was the development of an accurate forecasting

[2] Peak season refers to the period of significantly increased demand for shipping and delivery services, typically occurring during major shopping events and holidays.

system. This was essential to precisely estimate daily workforce requirements, a key factor in ensuring operational efficiency and avoiding both understaffing and overstaffing. Building on the improved forecasting, the VET FLEX model was designed to address intra-week ability gaps by optimizing voluntary extra shifts accepted by associates. This model provides dynamic, granular adjustments to align workforce allocation with fluctuating operational demands, ensuring that UTR capacity is maximized while minimizing volume losses.

The forecasting system played a foundational role, offering reliable predictions that informed and enhanced the VET FLEX model. Accurate forecasts of daily workforce needs were critical to aligning staffing with operational demands, reducing inefficiencies, and achieving cost savings. Furthermore, the integration of Machine Learning methods in the VET FLEX model ensures continuous adaptation to real-world changes through a feedback loop, further enhancing operational precision.

Together, the forecast and VET FLEX models represent a significant leap forward in planning efficiency. These innovations not only optimize workforce allocation but also deliver substantial financial savings, with projections for 2024 and 2025 showing reduced costs and increased acceptance rates for voluntary shifts. This highlights the pivotal role of data-driven solutions in transforming supply chain management and meeting the demands of a dynamic operational environment.

Chapter 3
AMAZON VOLUME FORECASTING SYSTEM

3.1 **Introduction to Amazon Forecasting System**

In this project, the focus of the forecasting process was specifically on volume prediction. Accurately estimating the volume of packages to be handled in various shifts is critical for ensuring that the right number of associates is available to meet demand. Within Amazon Logistics, volume forecasting is one of the most discussed and critical topics due to its immense complexity. Amazon employs three main types of forecasting, which are further divided into more specific analyses:

- **Short-term forecasting**, which covers the immediate 0–12-week period, is the most precise and detailed of all forecasting types at Amazon. It is fundamental for day-to-day operations and requires extreme accuracy since it directly impacts immediate operational decisions. The system leverages real-time data and sophisticated machine learning algorithms that consider countless variables, from current inventory levels to in-transit shipments and immediate demand signals. This short-term forecast is crucial for daily labor planning, helping determine the exact number of associates needed for each shift across various functions. It analyzes expected volume patterns throughout the day and week to ensure optimal workforce scheduling. Additionally, it plays a vital role in inbound planning, guiding decisions about receiving capacity and dock door scheduling, and outbound planning, including determining the number of trucks needed for delivery and optimizing delivery routes. The process is highly dynamic and constantly updated, considering real-time sales data, historical hourly and daily patterns, weather forecasts, local events, marketing promotions, website traffic patterns, and even customer search trends.

- **Medium-term forecasting**, which spans the 3–12-month period, focuses on operational planning and resource allocation. This forecast heavily relies on historical sales data, seasonality patterns, growth trends, and planned events like Prime Day or Black Friday. It is essential for capacity planning, staffing decisions, inventory purchasing, and budget allocation. The system considers market trends, planned marketing initiatives, competitor activities, economic indicators, new product launches, and changes in the marketplace to generate accurate predictions.
- **Long-term forecasting** looks further ahead, typically 1-3+ years, and guides strategic decision-making and infrastructure investments. This type of forecasting incorporates macroeconomic trends, company expansion plans, e-commerce market growth projections, competitor analysis, and anticipated technological innovations. These forecasts influence major strategic decisions about new fulfillment center locations, infrastructure investments, long-term resource planning, and market strategy development.

3.2 **Forecasting in the VET Flex Project context**

In the context of the VET Flex project, we utilized a specific branch of short-term forecasting known as Weekly Forecasting, designed to predict package volumes over a 0 to 6-day horizon. However, this type of forecasting presented a notable limitation for our project: it was overly focused on volume predictions and lacked consideration for Amazon's operational aspects. To address this issue, and with the support of the Forecast team, we have decided to expand the model's scope. Instead of exclusively predicting volumes and related metrics, we enhanced it to include additional operational metrics derived from the volume forecast. To achieve this, the forecasting system was designed with three fundamental components. First, a volume forecasting system based on Prophet provided accurate daily predictions of package counts for each shift and cycle. These volume forecasts were then used to calculate the opportunity headcount, representing the number of additional associates required to handle the forecasted workload. For example, if the predicted volume for a given day was 80,000 packages, approximately 180

associates would be needed. If only 160 associates were scheduled for that shift, this would result in 20 opportunity headcounts. Once these positions are offered and accepted by associates, they become VET Headcount.

This step was particularly critical, as prior to the VET Flex Project, the VET allocation process was managed using a simple, heuristic-based threshold. For instance, using the same example of 80,000 packages and 160 scheduled associates, requests would be made for an additional 6.5% of the total number of associates in the shifting this case, approximately 12 associates. However, this approach had two major flaws. First, requesting 12 associates would fail to meet the operational requirement of 180 total associates, leaving the shift understaffed. Second, even in scenarios where 12 additional associates might suffice, there was no mechanism to ensure that the acceptance rate would yield the precise number of workers required. The lack of precision in both volume forecasting and workforce planning often resulted in inefficiencies, either through under-staffing or over-staffing. By integrating the volume forecasting process with an operationally focused model, the VET Flex Project resolved these issues. This innovative approach ensured that workforce requirements were accurately calculated based on operational needs, aligning VET requests with predicted acceptance rates to optimize staffing levels effectively.

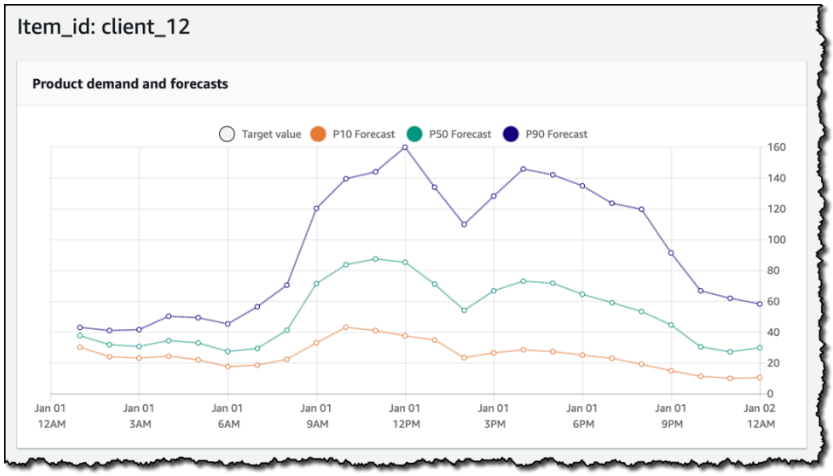

Figure 2: Sample output of product demand forecast from Amazon Forecast Tools

3.3 **Forecasting process**

To achieve the results outlined in the VET Flex Project, a robust forecasting framework was implemented that seamlessly combined advanced forecasting techniques with operational logic. Before delving into the technical details, it is essential to acknowledge that various initiatives within Amazon—such as new associate programs, Amazon Robotics processes, and other logistics-related innovations—played a critical role in shaping the forecasting process. However, since these initiatives are not yet publicly disclosed, we are unable to discuss them in detail, despite their significant contributions to the project's success. The operational forecasting component of the project was built upon an already solid volume forecasting system, which leveraged the capabilities of Prophet. This system had demonstrated its effectiveness in accurately predicting daily package volumes for each shift and cycle. Our primary objective was to enhance these existing volume predictions by integrating them with key operational metrics, thereby enabling more precise and dynamic workforce planning.

To achieve this, we utilized several critical metrics, including Units per Hour (UPH), Average Time per Task (ATT), and Customer Promise Attainment (CPA). These metrics, along with others, allowed us to refine the forecasting process by providing granular insights into the number of associates required in each section of the warehouse. This not only improved accuracy but also reduced the workload for Program Managers, who previously had to manage these workforce planning aspects manually.

By embedding operational logic into the volume forecasting system, we achieved a significant leap forward for the VET project. This integration not only optimized workforce planning but also uncovered new opportunities for innovation in logistical forecasting. These opportunities hold great potential for further exploration, paving the way for advancements that could transform how logistical challenges are approached in the future.

Chapter 4
DATA PREPROCESSING AND MODEL OPTIMIZATION

4.1 **Data Preparation and AWS S3 Buckets**

After conducting an accurate forecasting process, the next step is extracting the necessary data to build our Machine Learning model, which will enable us to achieve our final goal. Thanks to the services offered by Amazon, particularly AWS, data extraction is both fast and easy to implement. This is made possible by leveraging S3 buckets.

> «An **Amazon S3 bucket** (Simple Storage Service) is a fundamental building block within Amazon Web Services (AWS), providing a scalable, secure, and highly durable object storage solution. At its core, an S3 bucket acts as a virtual container where data objects—such as files, images, videos, or backup are stored. The design philosophy behind S3 is simplicity and reliability, making it a preferred choice for developers and organizations needing to store and retrieve vast amounts of data. » (Amazon, 2024)

An S3 bucket functions similarly to a directory in a traditional file system, but with enhanced capabilities suited for cloud storage. Each bucket is uniquely named across all of AWS, ensuring global uniqueness and accessibility. Buckets provide a way to group objects and apply policies or configurations at a bucket level, such as access permissions, version control, and lifecycle management. Uploading or importing data into an S3 bucket is a straightforward process and can be done in several ways. The AWS Management Console, an intuitive web interface, allows users to manually upload files by simply dragging and dropping them into the bucket. For programmatic access, AWS provides a rich suite of tools, including the AWS Command Line Interface (CLI), SDKs for various programming languages like Python (using the `boto3` library), Java, or Node.js, and APIs. Since the project has been developed in Python, `boto3` library was used for data preparation.

To start the Machine Learning part of the VET project, the first step involved gathering and structuring the necessary data for analysis. We retrieved two key datasets directly

from an S3 bucket: the first containing data related to acceptance rates and the second providing planning information, referred to as the "alps plan." These files were transformed into structured tabular formats using the Pandas library to enable efficient processing and in-depth analysis. The purpose of this process is to ensure that the data is accessible and ready for use in subsequent phases of the project, where it will be analyzed and modeled to generate valuable insights. Using S3 as the storage solution ensures the scalability needed to handle large datasets, while the integration with tools like Pandas allows for a seamless transition from raw data to actionable formats, then thanks to Python we will be able to explore the data and manipulate them.

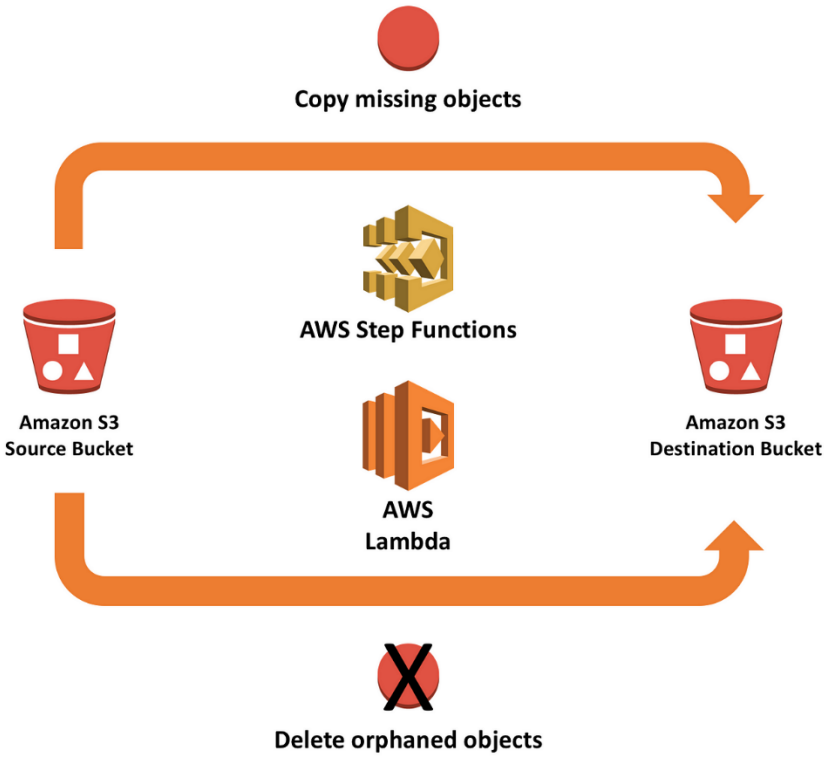

Figure 3: AWS architecture diagram showing S3 bucket synchronization workflow.

4.2 **Data Exploration**

Before starting any kind of Machine Learning process, it is particularly important to understand the data at hand. To do so, we must understand all the columns that we are going to find in our datasets and how we are going to use them.

In the alps plan dataset, we can find information about the operational part of every VET, with information such as:

Variable Name	Description	Data Type
site	Name of the delivery station	*String*
ofd_date	Date of the request for the VET	*Date*
shift_date	Date when the shift cycle begins	*Date*
activity	Type of activity the associate is expected to perform	*String*
Opportunity_type	Type of opportunity created	*String*
cycle	Name of the cycle during which the VET will be operated	*String*
sub_activity	Secondary activity to be performed during the VET shift (if present)	*String*
country	Country where the delivery station shift takes place (only England in this dataset)	*String*
volume	Volume of work to be handled during the shift	*Numeric*
rostered_hours	Number of hours pre-scheduled for the shift	*Numeric*
absence_hours	Number of absence hours recorded at the time of the VET shift request	*Numeric*
vto_hours	Voluntary Time Off hours	*Numeric*
available_headcount	Number of employees available for the shift or operational cycle	*Numeric*

vto_hc	Number of employees who opted for Voluntary Time Off	*Numeric*
vet_hc	Number of employees who chose to work extra hours voluntarily	*Numeric*
show_hours_needed	Number of hours required to complete the planned work	*Numeric*

Figure 4: Name, Description and Type of the ALPS Plan Data frame

In the acceptance dataset we can get information about the VET. We have some common columns such as *ofd_date, site_id, opportunity_type,* that will be particularly useful later in the project to make some join between the two datasets. The other columns are:

Variable Name	Description	Data Type
absence_hours	Number of absence hours recorded at the time of the VET shift request	*Numeric*
opportunity_instant_vto	Whether the opportunity is an immediate voluntary time-off (VTO) offer	*String*
opportunity_shiftend	End time of the opportunity's shift	*Datetime*
opportunity_shiftstart	Start time of the opportunity's shift	*Datetime*
opportunity_signupstart	Starting time/date when workers can sign up for the opportunity	*Datetime*
opportunity_headcount	Total number of workers required for the opportunity	*Numeric*
accepted	Number of workers who accepted the opportunity	*String*

time_accepted	Time at which the opportunity was accepted	*Datetime*
country_code	Country code associated with the site or opportunity	*Numeric*
direct_indirect	Whether the opportunity is for direct work (production tasks) or indirect work (support functions)	*String*

Figure 5: Name, Description and Type of the Acceptance Data frame

The final aim is to create a third data frame by merging these two that contains the most essential information concerning the shifts and the acceptance rate for each of them.

4.3 **Data Pre-Processing and Data cleaning**

Data preprocessing and cleaning are fundamental steps in any Machine Learning project. In our case, preprocessing was carried out at two distinct stages. The first and more extensive phase was conducted at once after obtaining the data frame, ensuring the data was organized in the most useful and proper way. Next preprocessing steps were performed before applying each Machine Learning algorithm, tailored to the specific requirements of the task. This approach was necessary because the quantity and type of data needed differ depending on whether a classification or prediction algorithm is used, as well as the library chosen. The preprocessing strategy was consciously designed to accommodate the use of either a forecasting algorithm or a binary classification model. The latter was aimed at predicting whether the acceptance rate would be below (0) or above (1) 80%.

As a first step, we converted specific columns having temporal information into a standardized datetime format to ensure uniformity across the dataset. We then filtered the "Opportunity_type" column to keep only entries representing VET, as the project focuses exclusively on these. Additionally, we dropped all opportunities lasting less than 30 minutes, as these stood for a different type of VET that is more immediate and managed

autonomously by Amazon sites, and thus outside the scope of our analysis. Further filters were applied to remove potential biases from the `cycle` and `shift` columns. Finally, we performed standard checks to find and address null values and ensured all columns were converted to the correct format.

Then, we proceeded to merge the two data frames, *alps_plan* and *acceptance*. For this purpose, a third data frame named *merged_df* was created, using the right merge on the columns 'site', 'ofd_date', and 'cycle'. This allowed us to obtain a new data frame containing both the data related to employee shifts and the acceptance rate, which would later be fed into the Machine Learning model that we planned to develop. Now, we have reached one of the most critical steps in the entire preprocessing phase, where a critical step of data processing has been performed to categorize the time difference between the opportunity signup start time and the shift start time into predefined categories based on business logic. This categorization is essential for analyzing and interpreting when workers engage with opportunities, relative to their start times, and for optimizing workforce planning.

First, the relevant columns, opportunity_signupstart and *shift_start_datetime*, were converted into a standardized datetime format to ensure consistent manipulation of temporal data. This step ensures that all future operations can accurately compare and calculate differences between timestamps. A function named *categorize_time_diff* was then defined to assign each row in the dataset to one of several time categories. These categories, ranging from "D-7" (7 days before the opportunity shift start) to "During Shift," were derived based on specific rules that consider both the signup time, and the shift start time compared to the opportunity's operational forecast date (ofd_date). The logic implemented in the function involves the following:

1. Calculating specific time thresholds, such as "D-7" to "D-1 PM," using the ofd_date as the reference. These thresholds delineate daily and intraday intervals leading up to the shift start.

2. Other conditions distinguish between morning and afternoon signups for "D-2" and "D-1," ensuring granularity in categorization.
3. More refined rules find signups occurring within an hour of the shift's start as <1 HR Before opportunity start time.
4. Lastly, any signup time occurring after the shift's start is labeled as "During Shift," encompassing immediate opportunities or changes.

The function evaluates each row in the dataset, checking the signup start time against these calculated thresholds. Based on the comparison, the proper category is assigned to a new column, *time_diff_category*. To supplement this categorization, another column, *time_diff_hours*, was calculated. This column measures the exact time difference between the signup start time and the shift start time in hours, providing additional detail for temporal analysis. This numerical measure complements the categorical data by enabling precise comparisons and potential insights into signup behavior trends. Thanks to this step, we were able to analyze the data frame and understand when VET requests were given. As expected, most requests were concentrated between D-1 (the day before or the same day as the shift) and D-2 (two days before the shift). However, we were surprised to find that a massive portion of requests, accounting for 18% of the total, were made as early as D-7 (one week before). This happens because the maximum time allowed for giving a VET request is one week prior to the shift, meaning that even in the absence of immediate needs, the system must wait until the week leading up to the shift. From the data analysis, we observed that approximately 75% of the dataset consisted of requests made at D-1 and D-2, while D-7 accounted for the remaining 18%. Based on this information, we decided to filter the data frame, keeping only requests made within the time spans D-1, D-2, and D-7. Subsequently, D-7 was also removed to prevent the model from being influenced by requests submitted far in advance. This decision was made because the primary objective of the model is to offer precise insights into shifts with sudden staff shortages, requiring an immediate and accurate prediction of VET needs.

During the initial phase of developing our Machine Learning models, we noticed a significant lack of key information within the dataset. For example, there was no clear

sign of the number of associates needed for a specific shift. Instead, the VET data was recorded in hours, as it was originally designed for payroll purposes, reflecting the hours to be compensated rather than the operational headcount needed. Another crucial missing piece was the acceptance rate—the percentage of requests that were accepted. This metric was essential, as the number of accepted requests could often fall short of the demand or, conversely, exceed the operational need.

Although those data points were absent, we were able to derive them through mathematical formulas with the data in our data frame.

This process of extracting and engineering features became an essential step in ensuring that the models had the inputs needed to deliver accurate and actionable predictions After performing these derivations, the updated data frame was saved for use in subsequent Machine Learning algorithms. As the project progressed, we continued with further pre-processing activities, addressing additional data gaps that we successfully filled through calculations and derivations, which significantly improved the model's performance. One of the first steps involved applying a filter followed by aggregation using a `*group by*` operation, grouping all rows with the same cycle and date. This decision was driven by the presence of multiple entries in the data frame associated with the same cycle but referring to different activities. Since the client's request focused on obtaining a general acceptance rate, based on shifts or cycles, we decided to aggregate the data to provide a comprehensive view. Subsequently, we developed new parameters to enhance the model's performance.

One of the first calculated was the acceptance rate in percentage, derived using a straightforward and intuitive formula.

$$\boldsymbol{acceptance\ rate}\ (\%) = \left(\frac{opportunity\_headcount}{accepted}\right) \times 100$$

Next, we calculate a fundamental metric to understand the actual operational capacity and compare it with the planned capacity. This metric is crucial for improving resources and workforce planning in the future. The formula used to derive this value is:

$$\boldsymbol{hc\_on\_site\_hour} = rostered\_hours - absence\_hours + vet\_hours - vto\_hours$$

We start with the planned hours, represented by *rostered_hours,* which indicate the total hours employees were expected to work. However, not all these hours are covered due to various absences, both planned and unplanned, captured under *absence_hours.* At the same time, there are cases where employees voluntarily choose to work added hours through the Voluntary Extra Time program (*vet_hours*). These hours increase the total number of available working hours. Conversely, some employees may decide to take time off voluntarily, using the Voluntary Time Off program (*vto_hours*), which reduces the overall availability. By adding and subtracting these components, we derive *hc_on_site_hours,* which represents the final count of hours employees worked or were available to work on-site. This metric provides a clear picture of the workforce's effective contribution during a given period, allowing for better alignment of operational expectations and resource management.

To further enhance the model's performance, we decided to calculate the duration of each shift, as this parameter could significantly influence an associate's decision to accept or decline a voluntary extra work shift. To achieve this, we split the original columns shiftenddatetime and *shift_start_datetime*, which contained both the date and time of the shift's start and end, into four new columns: *Shift_start_date, Shift_end_date, Shift_start_time, and Shift_end_time*. This allowed us to separate the information related to the date and time of each cycle, making the data more granular and easier to analyze. Once the new columns were created, we calculated the duration of each shift in hours and minutes by computing the difference between the start time *(Shift_start_time)* and the end time *(Shift_end_time).* This step provided a key parameter, *Shift_duration_hours*, representing the total duration of each shift.
Subsequently, as the last step of pre-processing related to creating new variables, we calculated two added critical metrics: *Requested Headcount* and *Rostered Headcount.*

$$\boldsymbol{Requested\_Headcount} = \frac{Shift\_duration\_hours}{hc\_on\_site\_hours}$$

This metric stands for the number of employees required to cover the available working hours during each shift. It was calculated by dividing the on-site working hours (hc_on_site_hours) by the shift duration in hours (Shift_duration_hours). This calculation provides a correct estimate of the workforce needed to handle the actual working hours for each shift.

$$rostered\ headcount = \frac{rostered\_hours}{shift\_duration\_hours}$$

Similarly, this metric stands for the number of employees scheduled to work during the shift, based on its duration. It was obtained by dividing the planned hours (rostered_hours) by the shift duration in hours (Shift_duration_hours). This value offers a measure of the workforce that was originally planned, which is helpful for comparing projections with actual operations. Integrating these metrics into the model allows for precise analysis and comparison of workforce availability and planning, thereby improving the quality of predictions and resource allocation.

4.4 **Explorative Data Analysis and Correlation Matrix**

One of the most common and valuable practices in the field of data science is Exploratory Data Analysis (EDA). EDA represents the process of exploring and analyzing datasets to gain meaningful insights, uncover underlying structures, identify key variables, detect outliers and anomalies, test assumptions, develop models, and determine the best parameters for future predictions. *(Suresh Kumar, 2020)* In our case, EDA played a crucial role in helping us identify the most appropriate types of models for our analysis. This was achieved through visualizations that revealed the distribution of data and the relationships between variables, the latter analyzed using a Correlation Matrix. As a first step, we analyzed the distributions of the variables to better understand

their structure and identify patterns, anomalies, or trends. Beyond the distributions, we also analyzed the temporal patterns present in the data. By visualizing trends over time, we discovered that periods with higher order volumes coincided with significant increases in voluntary extra time (VET) requests. These insights prompted us to consider incorporating forecasting models alongside binary classification models, enabling us to account for seasonality and predict workforce demands more effectively.

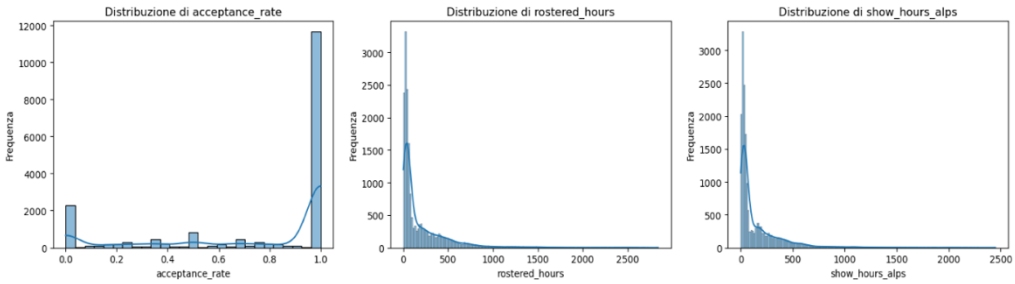

Figure 6: Distribution of acceptance rate, rosterd hours and show hours alps columns.

Another significant observation emerged from analyzing the distribution of *Shift_durations_hours*, which stands for the duration of a single shift. This analysis revealed that most voluntary shifts lasted either 6 or 12 hours, highlighting a discrepancy with the largest allowed shift duration of 10 hours. Upon further investigation, we discovered that the data collection system records shift as 12 hours when a 4-hour shift is worked on the same day as an 8-hour shift. For instance, if an associate works an 8-hour shift and, on the same day, accepts a voluntary VET shift of 4 hours, the system erroneously combines them into a single 12-hour shift. In contrast, if the voluntary shift is accepted for the following day, it is correctly recorded as a 4-hour shift. To address this issue and cut outliers in the dataset, we corrected all 12-hour shifts by reclassifying them as 4-hour shifts, accurately reflecting the true duration of the voluntary shift.

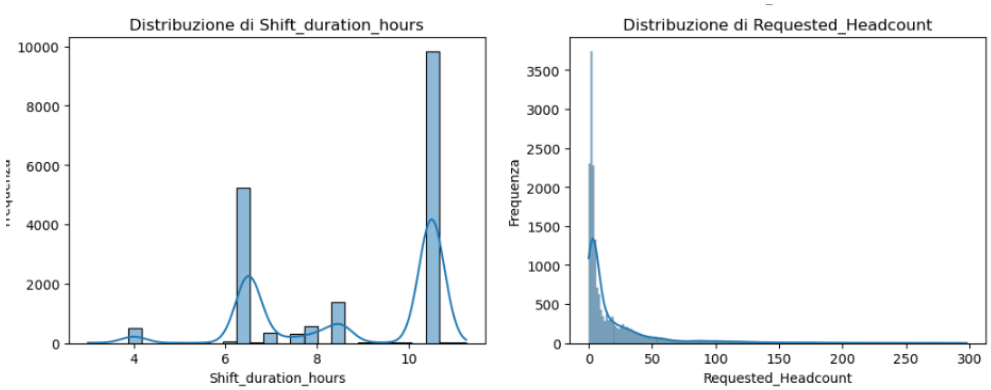

Figure 7: Distribution of Shift Duration and Requested Headcount.

As the next step we performed a correlation matrix: "A correlation matrix is a symmetric matrix that displays the correlation coefficients between pairs of variables in a dataset. Each element in the matrix quantifies the degree to which two variables are linearly related, with values ranging from -1 (perfect negative correlation) to +1 (perfect positive correlation). The diagonal elements are always 1, indicating a perfect correlation of each variable with itself. Correlation matrices are essential in multivariate statistical analyses, as they help in understanding the relationships between variables and in identifying patterns within the data." (Hadavand-Siri & Deutsch, 2012)

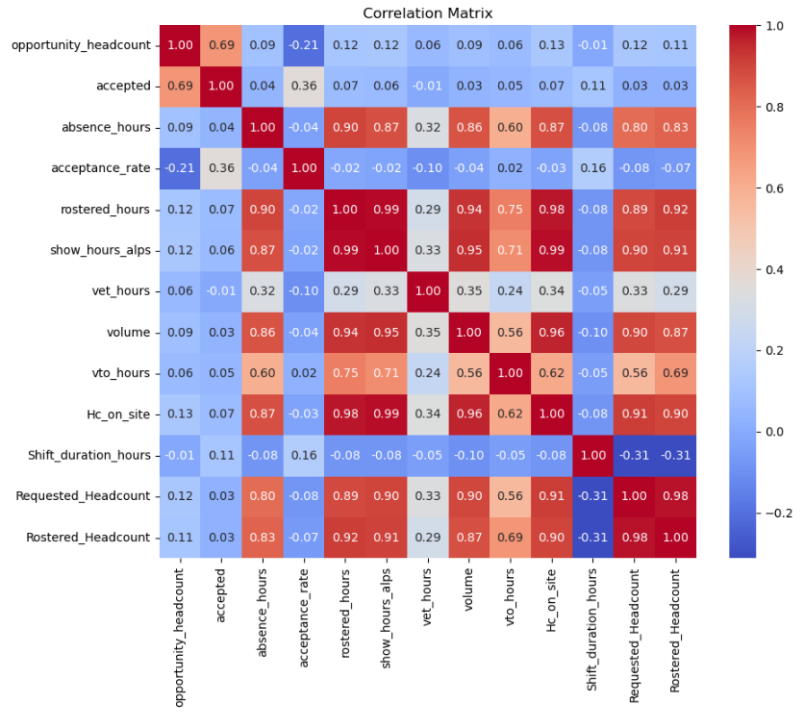

Figure 8: Correlation Matrix of VET Flex Project Variables.

Through the analysis of the correlation matrix of our data frame, we observed that the variable accepted showed an incredibly low correlation with most of the other variables in the dataset. While this might initially seem problematic for the model's performance, it is not necessarily a cause for concern. In fact, highly correlated variables can often present greater challenges, such as multicollinearity, which can negatively affect the stability and interpretability of a model. A low level of correlation does not automatically imply poor predictive power, as certain variables might still hold valuable information when combined with others through non-linear relationships captured by the model. To enhance the model's ability to capture temporal patterns and improve its understanding of past influences on current outcomes, we decided to create lag variables. Lag variables, as defined in academic literature, are features derived by shifting the values of a time-dependent variable backward by one or more periods. These features introduce a temporal context into predictive models, allowing them to leverage historical information. In our case, we created lag variables such as lag accepted (representing the number of acceptances recorded in the previous shift), lag opportunity (indicating the opportunities

available in the past), and lag_absence_hours (reflecting the hours of absence in previous shifts). These lag variables allowed the model to better understand how historical trends and events influence the likelihood of future acceptances, ultimately improving its predictive performance.

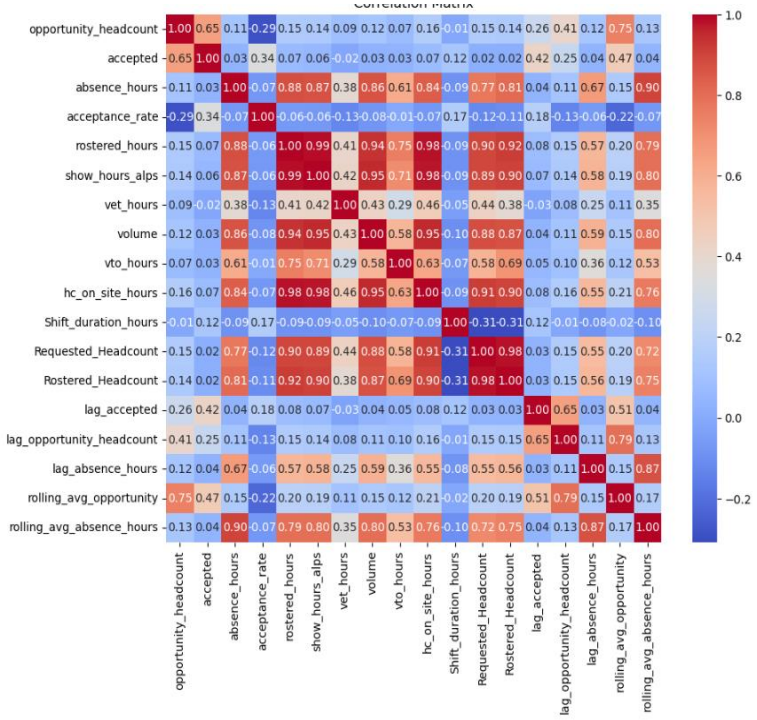

Figure 9: Correlation Matrix of VET Flex Project with Lag and Rolling averages.

The introduction of lag and historical variables significantly improved the model's performance. Thanks to these variables the model became more able to capture local trends and it highlighted lagged relationships between inputs and outputs, such as the potential impact of a small number of acceptances in the earlier shift on the current one.

4.5 **How we approached the Machine Learning problem**

As previously mentioned, our work on the Machine Learning component began with a meticulous pre-processing phase.

The first critical decision was to determine the overall methodology: reducing the problem to binary classification rather than employing traditional regression or forecasting techniques. To understand this choice, it is important to start with the fundamental question: What are we trying to achieve? The goal was to predict the percentage of people willing to accept an extra voluntary shift (the acceptance rate) without overestimating or underestimating the number of requests, thus avoiding overstaffing or understaffing relative to the expected volume.

For example, if we know that seven people are required for a specific shift, we cannot afford to request 20 (causing unnecessary strain) or limit ourselves to exactly seven, given that the acceptance rate is rarely 100%. Therefore, a margin of safety is necessary: in this case, requesting eight or nine individuals, knowing that the average acceptance rate is around 90%. While this process may seem straightforward, it operates within a complex framework influenced by a fundamental principle within Amazon: standardization. Standardization is essential for ensuring operational consistency within a single country and across regions. It allows the company to respect local legal requirements, establish clear and uniform limits for each warehouse, maintain objective metrics to evaluate the performance of different warehouses, and avoid organizational discrepancies. Without standardization, there would be a risk of creating ad hoc rules for each warehouse, generating inefficiencies that could compromise the stability of the supply chain at a regional level. Once the operational requirements were defined, we focused on identifying the most suitable approach for our model. Traditional regression would have been ineffective for several reasons, the most significant being the nature of the target variable, the acceptance rate. This variable is almost binary: in 80% of cases, it is either 0% or 100%, with only 7% of cases falling between 1% and 75%. Given this skewed distribution of the output values, a regression model might strive in accurately predicting the lesser frequent values (i.e., the ones in the range 1-75%).

Considering the above, we opted for a binary classification approach, defining a threshold for the acceptance rate that was deemed "positive" (the 1 in binary classification) in collaboration with operational teams in the warehouses. This threshold was set at 80%,

with a more aggressive approach compared to the past, to better align with operational needs. Furthermore, the number of extra workforce requests (flex-up) was set within a range of 8% to 12%, varying by country and based on historical data. For example, in Europe, the previous cap was set at 6.5%, but our analysis demonstrated that a broader range was necessary to better manage demand fluctuations. To validate the effectiveness of our model, we conducted simulations on various flex-up levels using a dataset that exclusively contained data from warehouses in the UK. The goal was to identify the ideal number of extra workforce requests that would maximize coverage without overloading the system. The results showed that, under exceptional circumstances or during extraordinary events, it is theoretically possible to achieve a flex-up of 30% with an acceptance rate of 95%. However, this scenario is not realistic for day-to-day operations, as the maximum historical flex-up recorded in the dataset (over two years) was 17%, which occurred only three times. This highlights the importance of keeping requests within realistic ranges, balancing operational needs with system sustainability. Thanks to this structured approach, we were able to transform a complex challenge into a practical and applicable system. The combination of a targeted method, historical data analysis, and engagement with operational teams allowed us to optimize extra workforce requests, improving overall efficiency without compromising sustainability. This project not only provides immediate solutions but also establishes a replicable framework for future optimizations within Amazon's supply chain

4.6 **Binary Classification: Concepts and Metrics**

Classification problems can be categorized into binary, multiclass, and multilabel tasks. In binary classification tasks, only two classes are considered, which are commonly referred to as the positive and negative class: for example, healthy vs. diseased, under expressed vs. overexpressed, smoker vs. nonsmoker, or in our case high acceptance vs low acceptance.

In the context of binary classification, each observation $\boldsymbol{x_i}$ is associated with a target variable $\boldsymbol{y_i}$, which takes the value 1 for the positive class (in our case, "high acceptance") and the value 0 for the negative class ("low acceptance"). To evaluate the effectiveness

of the model, a confusion matrix is used, which compares the model's predictions with the actual values. The matrix includes:

- **True Positives (TP),** representing cases of high acceptance correctly identified.
- **False Positives (FP),** representing cases of low acceptance incorrectly predicted as high acceptance.
- **False Negatives (FN),** representing cases of high acceptance not recognized by the model.
- **True Negatives (TN),** representing cases of low acceptance correctly identified.

From the confusion matrix, several key metrics are derived to quantify the model's quality. One such metric is *accuracy*, which represents the overall proportion of correct predictions:

$$Accuracy = \frac{(TP + TN)}{(TP + FP + FN + TN)}$$

However, in our case, accuracy is not sufficient, as it does not differentiate between the importance of errors for the two classes. For instance, erroneously predicting a high acceptance case as low acceptance (FN) might have a greater operational cost than predicting a low acceptance case as high acceptance (FP). For this reason, metrics such as *precision* and *recall* become crucial. Precision measures the reliability of the model's positive predictions:

$$precision = \frac{TP}{TP + FP}$$

This is particularly important if we want to minimize cases of low acceptance being incorrectly classified as high acceptance. On the other hand, recall measures the model's ability to correctly identify high acceptance cases:

A high recall ensures that the model captures most of the high acceptance scenarios, which is critical in our context. Since precision and recall represent two potentially conflicting objectives (improving one might reduce the other), a composite metric like the *F1-score* is useful to balance these two aspects:

$$F1\ Score\ = 2\ \times \frac{precision\ \times\ recall}{precision\ +recall}$$

The F1 score is particularly relevant in our case, as it provides a harmonic balance between the model's precision and its ability to correctly identify positive cases.

4.7 **Forecasting Models for VET**

As part of our initial approach, we considered leveraging forecasting models to predict relevant metrics for the VET project. However, these models did not perform as expected. While we adopted an iterative testing approach throughout the project, allowing us to experiment with a variety of models to identify the most suitable one—we decided to halt further exploration of forecasting models early. This decision was not because we immediately concluded that forecasting was unsuitable but rather because the preliminary results indicated significant limitations for our specific context.

Our first attempt at forecasting employed Prophet, a widely used model in corporate analytics. As Meta describes it:

"*Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend and typically handles outliers well."*

Prophet's strength lies in its ability to model seasonality and its interpretability. These qualities made it an attractive option for VET, where the final output needed to be easily

understood by stakeholders. In the supply chain domain—especially for a large-scale e-commerce operation like Amazon—accounting for seasonality is critical. Specific periods, such as Black Friday, Christmas, or Prime Day, result in significant surges in volume, which, in turn, increase the demand for an additional workforce. Prophet's ability to integrate these recurring seasonal peaks and its flexibility in handling holiday-specific effects appeared to align well with our goals. Unlike traditional forecasting models, Prophet requires only a minimal set of variables for implementation. These include:

- **ds**: The column representing the date or timestamp.
- **y:** The column representing the target variable to be predicted
- **lower window and upper window** (optional): Parameters define the time range (in days) around a given event to consider its influence.

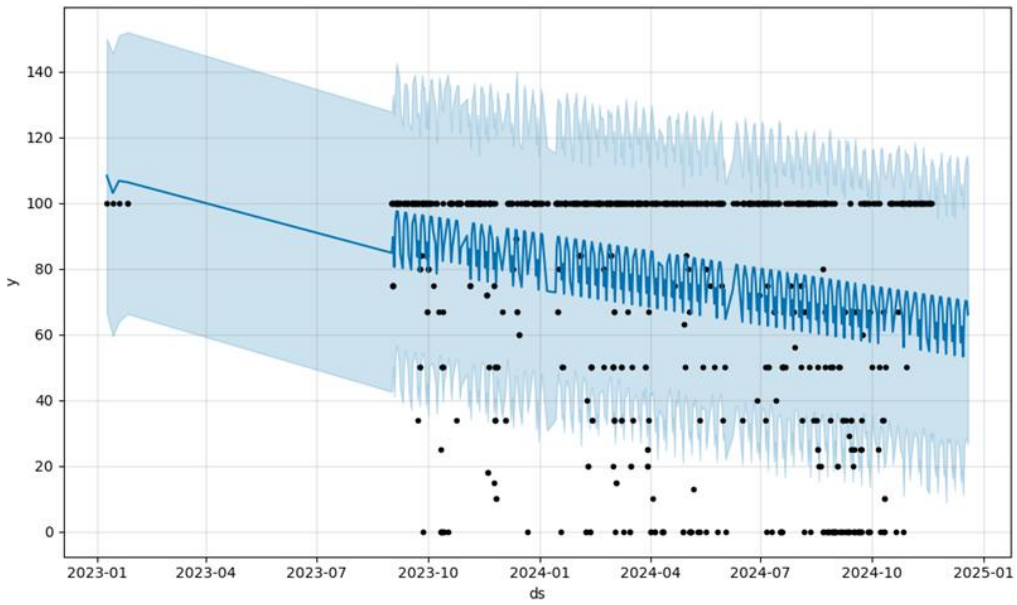

Figure 10; Prophet results from VET Test

To evaluate the model, we employed the standard metrics for time series forecasting, namely MSE, MAE, and RMSE. However, the results returned values that were too low to be deemed acceptable. This prompted us to consider the fact that Prophet performs well with linear patterns or regular seasonality but struggles to handle complex or non-linear relationships. Consequently, we concluded that Prophet was not suitable for

identifying VET, as the relationships in our context are complex and depend on multiple variables.

After the test with Prophet, before giving up with Forecasting we tested SARIMA: a well-suited model for data with regular seasonal patterns, and its parameterized approach allows for fine-tuning to better model complex relationships. *(Permanasari, Hidayah, & Bustoni, 2013)* For the VET project, understanding demand surges and workforce requirements necessitated a model capable of accounting for not just seasonality but also short-term dependencies and trends. At its core, SARIMA captures the relationships within a time series by combining regular (non-seasonal) and seasonal components. These components allow the model to account for short-term dependencies as well as long-term cyclical behaviors. For instance, in data with annual seasonality, SARIMA can model how values at the same time last year influence current observations. SARIMA's flexibility comes from its parameterized structure, which balances simplicity with its ability to capture complex patterns. By adjusting these parameters, SARIMA can adapt to a variety of datasets and forecasting scenarios, making it a versatile tool in time series analysis. The core components are

Differencing (d, D):

a. Differencing removes trends and seasonal patterns to make the series stationary. Non-seasonal differencing (*dd*d) handles overall trends, while seasonal differencing (*DD*D) addresses recurring patterns.

2. **Autoregressive (AR) Terms (p, P):**
    a. These parameters model the influence of past values on the current observation. For example, *p=1p = 1*p=1 means the current value is influenced by its immediate predecessor, while *P=1P = 1*P=1 considers past values from the same point in previous seasons.
3. **Moving Average (MA) Terms (q, Q):**

a. These terms smooth out fluctuations by using past forecast errors. For instance, $q=1q = 1$q=1 adjusts predictions based on the most recent error, and $Q=1Q = 1$Q=1 incorporates seasonal errors.

4. **Seasonal Period (s):**
    a. This parameter defines the length of the seasonal cycle. For example, $s=12s = 12$s=12 for monthly data or $s=7s = 7$s=7 for weekly data.

Once trained, SARIMA showed significantly better performance than Prophet. It effectively captured the seasonal patterns and provided forecasts that aligned more closely with real-world trends.

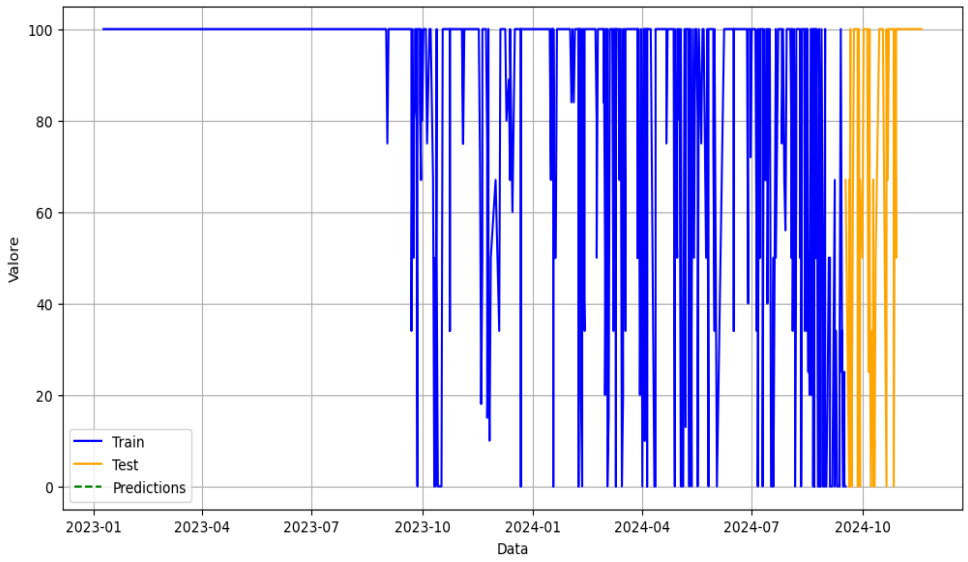

Figure 11: SARIMA Results from VET test

However, while SARIMA proved effective in modeling seasonality, we noticed it struggled with some of the non-linear and interaction effects present in the data. For instance, the relationship between site-specific characteristics, workforce availability, and external factors like unexpected demand surges was too complex to be fully captured by the model. Aware of these limitations, we decided to complement SARIMA with added models to decide if performance could be further improved. Specifically, we integrated a Random Forest model and a Neural Network. These models were selected for their

strengths: Random Forest excels at managing non-linear relationships and complex interactions, while Neural Networks are powerful at learning intricate patterns in high-dimensional datasets. Despite our efforts, neither model provided significant enough improvements to justify their added complexity. The Random Forest model, while capturing some non-linear effects, struggled with seasonality, which required additional preprocessing steps. Similarly, the Neural Network, although theoretically effective, proved difficult to optimize for our specific context. Since the results did not satisfy us, we decided to stop using classical forecasting models, as the quality of the data was not suitable for the proper functioning of these models.

4.8 **Classification Models**

A classification model is a predictive modeling technique that assigns discrete labels to input data by learning from a set of labeled examples. It works by finding patterns or relationships within the input features to accurately categorize new, unseen instances. *(OpenAI, 2024)* Classification involves learning a function that maps an input attribute set to one of the predefined class labels. Classification involves learning a function that maps an input attribute set to one of the predefined class labels. These models are widely used in various domains such as spam detection, medical diagnosis, fraud detection, and customer segmentation due to their ability to handle structured and unstructured data efficiently.

4.8.1 **Logistic Regression**

We decided to use logistic regression as it is quite a simple model to implement and interpret. For instance, in the VET project, logistic regression highlighted the most influential factors driving acceptance rates, such as the timing of shifts or the headcount availability. However, we must consider the fact that Logistic regression assumes a linear relationship between the independent variables and the log-odds of the dependent variable, which may not always hold true in real-world scenarios. Also, the model can struggle a lot with overfitting; to solve this problem we used some regularization techniques: In logistic regression, regularization is a method used to handle issues such as overfitting, which happens when the model performs very well on training data but

struggles with new, unseen data. This often occurs when the model gives too much importance to certain predictor variables, leading to overly complex and unreliable predictions. Regularization helps by adding constraints to the model, ensuring that it does not rely too heavily on any specific predictor. *(Hosmer, Lemeshow, & Sturdivant, 2013)* This makes the model more robust and better at generalizing new data. There are two main types of regularization used in logistic regression:

1. **L1 Regularization (Lasso)**: L1 regularization, also known as the Lasso (Least Absolute Shrinkage and Selection Operator), is a technique that adds a penalty equal to the absolute value of the magnitude of coefficients to the loss function in regression models. This method encourages sparsity in the model by driving some coefficients to zero, effectively performing variable selection and thus simplifying the model. *(Cortes, Mohri, & Rostamizadeh, 2012)*

2. **L2 Regularization (Ridge):** L2 regularization, commonly referred to as Ridge regression, involves adding a penalty equal to the square of the magnitude of coefficients to the loss function. This approach discourages large coefficients by shrinking them proportionally, leading to a more balanced model that can generalize better to unseen data. *(Cortes, Mohri, & Rostamizadeh, 2012)*

In the VET project, we used regularization to improve the logistic regression model for predicting the acceptance rate of voluntary shifts. L1 helped us with removing unnecessary variables such as “shift” and “rostered hours”, and thanks to that the model only relied on predictors that had a meaningful impact. Meanwhile L2 had no significant impact on the model since the dataset was already balanced and had a decent quality.

Class	Precision	Recall	F-1 Score
0	0.77	0.73	0.75
1	0.82	0.79	0.80

Figure 12: Results from Logistic Regression

4.8.2 **XGBoost**

XGBoost (Extreme Gradient Boosting) is a scalable and efficient implementation of gradient boosting machines, designed for speed and performance. It has been widely adopted in various machine learning competitions and real-world applications due to its ability to handle large datasets and complex models effectively. *(Guestrin & Chen, 2016)*
We decided to use XGBoost because it is an algorithm designed to handle large datasets efficiently. Moreover, it is more interpretable compared to other classification models, which is particularly helpful for reporting results to non-technical teams. However, XGBoost requires careful attention, as finding the best hyperparameters can be challenging. To address this, we used **grid search** to fine-tune the model: "Grid search is a simple yet effective approach to hyperparameter optimization that involves training and validating models over a grid of hyperparameter values, selecting the combination that minimizes the loss function or maximizes predictive performance." In the context of our project, we applied grid search to improve key hyperparameters of the XGBoost model, such as:

1. **Learning rate (η\etaη)**: Controls the step size for updating weights during training.
2. **Max depth**: Determines the maximum depth of each tree, balancing model complexity and overfitting.
3. **Number of estimators**: Specifies the total number of decision trees used in the ensemble.
4. **Subsample ratio**: Defines the proportion of the training data used for building each tree.

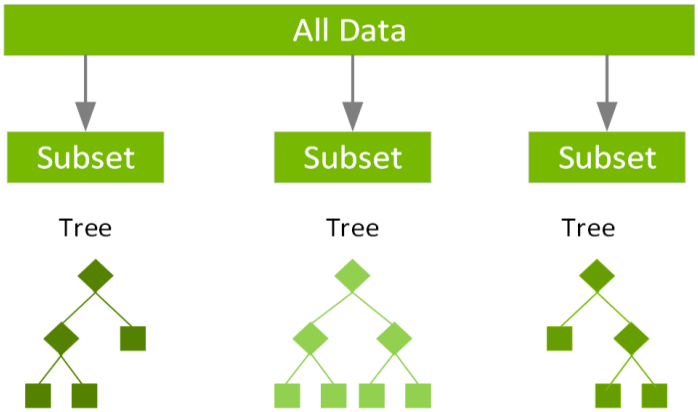

Figure 13 : Random Forest diagram showing three decision trees trained on different data subsets.

Thanks to the grid search, we were then able to better understand which parameters were best for our XG Boost and these were the results:

Class	Precision	Recall	F-1 Score
0	0.81	0.81	0.81
1	0.84	0.84	0.84

Figure 14: Results from XGBoost Classifier

4.8.3 **ADA Boost**

Adaptive Boost is a machine learning algorithm that ensemble method that combines multiple “weak learner” (simple classifiers, such as decision stumps) to create a "strong learner" capable of superior predictive performance. The core idea of AdaBoost is to assign higher weights to data points that were misclassified by previous models, thereby focusing the algorithm’s attention on harder-to-classify examples in subsequent iterations. *(Schapire & Freund, 1997)*

For our project, the AdaBoost Classifier represented a potentially strong choice due to its ability to combine different weak learners into a more accurate and robust strong learner. This characteristic allowed the model to progressively improve its performance by focusing on misclassified examples in each iteration. Moreover, AdaBoost is particularly

effective when working with categorical variables, such as the Shift and Cycle columns in our dataset. Its ability to handle these types of features made it well-suited for our data structure, where categorical variables played a significant role in influencing the acceptance rate. One of the key strengths of AdaBoost, which aligns with the other classification models we evaluated, is its interpretability. The results produced by the model are straightforward to understand, making it easier to find areas of improvement. Additionally, AdaBoost provides tools for analyzing feature importance, which was invaluable for conducting later feature engineering. By finding the most influential variables, we were able to refine the dataset and further enhance the overall performance of the model. To optimize the hyperparameters of the AdaBoost Classifier, we decided to use grid search, a systematic approach that allowed us to fine-tune parameters such as the number of estimators and the learning rate. While the model performed well overall, it did not reach the level of accuracy achieved by XGBoost. AdaBoost struggled with predicting cases where the acceptance rate was below 80%.

This limitation became evident in its difficulty recognizing instances where the acceptance rate was closer to zero, highlighting its slightly reduced sensitivity in such scenarios.

Class	Precision	Recall	F-1 Score
0	**0.77**	**0.78**	**0.78**
1	**0.83**	**0.86**	**0.85**

Figure 15: Results from ADA Boost

Although AdaBoost was a strong contender, its weaker performance on class 0 and lower sensitivity to lower acceptance rates made it less suitable than XGBoost for our project.

4.8.4 CAT Boost Classifier

CAT Boost (Categorical Boosting) is a gradient boosting algorithm specifically designed to handle categorical features efficiently without requiring explicit encoding. It operates similarly to XGBoost and AdaBoost but integrates automatic encoding mechanisms, making it particularly useful for datasets with many categorical variables.

In all other aspects, CAT Boost works very similarly to AdaBoost and XGBoost: it is an extremely fast model, easily interpretable, and can be improved through feature engineering. However, unlike the other models, CAT Boost proved to be inefficient and delivered poor results. One of the main limitations of the model is its handling of extreme cases, that is, when variables are either remarkably high or incredibly low. In a model like ours, which often had results equal to 0% or 100%, the model often did not correctly interpret these variables. Additionally, the "automatic" processing of categorical variables was not particularly necessary, as they could be easily optimized through one-hot encoding.

Class	Precision	Recall	F-1 Score
0	0.72	0.72	0.72
1	0.75	0.75	0.75

Figure 16: Results from CAT Boost Classifier

Despite its advantages, CAT Boost performed worse than XGBoost in our project. It struggled with extreme cases, failing to classify acceptance rates close to 0% or 100% accurately. Additionally, its automatic handling of categorical variables was unnecessary, as traditional encoding methods already worked well. Given these limitations, XGBoost remained the superior choice.

4.8.5 **Neural Network**

An *Artificial Neural Network* (ANN) is a computational model inspired by biological nervous systems, such as the brain, to process information. It consists of interconnected units, known as artificial neurons, which work together to solve specific problems. Thanks to their ability to learn from data, neural networks are particularly suited for tasks such as classification, pattern recognition, and decision-making. *(Sonali & Prinkaya, 2014)* An ANN is structured into three main layers of neurons:

- **Input Layer:** Receives data or input features and transmits them to the next layer for processing,
- **Hidden Layer:** Perform complex computations on the input data, extracting meaningful features and patterns
- **Output Layer:** Produces the networks prediction or outputs based on the processed information from the input layer.

The strength of ANNs lies in their remarkable ability to learn from data. During the training phase, the network is exposed to a set of labeled data, which is used to adjust weights and biases to minimize the difference between predictions and final outputs. This process is known as backpropagation. ANNs are therefore an excellent algorithm for a complex dataset like VET, as they can capture intricate relationships between variables. The only issue encountered with Neural Networks is, unsurprisingly, their slowness. Being a complex algorithm, ANNs require significant computational time, which posed a challenge for our project. Since the results needed to be updated multiple times throughout the day, this could become problematic, particularly when the project is scaled to a European level.

Class	Precision	Recall	F-1 Score
0	0.80	0.80	0.80
1	0.83	0.84	0.84

Figure 17: Results from Neural Network

4.8.6 **Voting Classifier**

A voting classifier is a machine learning model that gains experience by training on a collection of several models and forecasts an output (class) based on the class with the highest likelihood of becoming the output. (Ashish, 2023)To forecast the output class based on many votes, it averages the results of each classifier provided into the voting classifier. determine the output, the voting classifier averages the results provided by each participating model and uses most votes to determine the final class. Thanks to this process, it is possible to build a single model that learns from the results of several models, training them simultaneously.

There are two types of voting classifier:

- **Hard Voting**: The final class is the one with the absolute majority of votes. That is, the class with the highest probability of being predicted by our model. For example, in the case of VET we used XGBoost, ANN and ADA Boost Classifier. If in a row XGBoost and ADA predicted 1 and ANN 0, then the predicted class would be 1.
- **Soft Voting**: Here the predicted probabilities for each class are averaged, and the class with the highest average probability becomes the final prediction. For example, in VET, if XGBoost predicted 0.86, ANN 0.78 and ADA Boost 0.79, even though two predictions were below 0.80, the total average of them is 0.82, so the result would go in class 1.

In our study, we tested both Hard Voting and Soft Voting, but neither approach provided better results than XGBoost alone. The main issue arises from the fact that XGBoost is significantly better at distinguishing instances of class 0 compared to the other models. As a result, combining models through voting did not enhance performance but rather diluted the strengths of XGBoost in finding the minority class.

Hard Voting:

Class	Precision	Recall	F-1 Score
0	0.76	0.77	0.77
1	0.85	0.85	0.83

Soft Voting:

Class	Precision	Recall	F-1 Score
0	0.79	0.80	0.80
1	0.81	0.81	0.81

Figure 18: Results from Hard and Soft Voting classifier

4.9 **Model Output**

The project aims to generate daily forecasts for D-1 and D-2, the days when requests for VET (Voluntary Extra Time) are sent. On D-2, an initial request is made based on the available forecasts, while on D-1, when more accurate data on incoming package volumes at the fulfillment center is available, the request can be modified, although this rarely happens. To optimize this process, a system has been developed those extracts daily volume forecasts for D-1 and D-2 and uses an XGBoost model to determine the optimal number of VET requests. The algorithm iterates progressively up to a maximum of 50 requests, which represents the European limit set for 2025, and identifies the maximum number of requests where the predicted acceptance rate is at least 80%. This approach ensures that the optimal number of requests is identified, preventing both excessive requests, which might lead to unaccepted shifts, and underestimated requests, which could result in insufficient staff to handle the forecasted package volume. The final output, sent to Amazon teams responsible for workforce management in the Delivery Stations, is represented by the following sample table:

Country	Station	Shift	OFD Date	VET Request
UK	DBH3	PM	01/02/2025	3
SPAIN	ZAZ8	AM	01/02/2025	8
FRANCE	LYN2	AM	01/02/2025	13
SPAIN	BCN1	NS	01/02/2025	1
UK	LYS5	PM	01/02/2025	1

Figure 19 Output results sample

The output provides the country and warehouse, the shift for which the VET request is suggested, the shift date, and the maximum number of VET requests that can be made while ensuring that the predicted acceptance rate is equal to or above 80%. To ensure continuous improvement of the forecasts, the model is re-trained weekly with the most recent data collected week by week. Currently, the results show a gradual improvement over time, showing that the model is refining its predictive capabilities as new data is incorporated. In the future, as more detailed data becomes available and the quality of the information improves, we may also consider adopting a different or more advanced model that could further enhance performance in optimizing VET requests.

Chapter 5
FIRST RESULTS AND CONCLUSION

5.1 **Model Performance Evaluation and Optimization**

To evaluate the model's effectiveness, we systematically compare its predictions with real-world acceptance rates to ensure its reliability and accuracy. Over the last month, the average acceptance rate has increased by 22% compared to the months before the project's launch, showing that the model is having a positive impact on workforce planning. Currently, the acceptance rate stands slightly above 85%, showing a consistent improvement in the predictability of VET requests.

One of the key observations during this period is that the model has been more conservative than aggressive, meaning it has led to more cases where the suggested number of VETs was lower than what could have been realistically asked, rather than overestimating the need. This cautious approach has resulted in fewer instances of excess workforce allocation and more occurrences where additional workers could have been requested but were not.

Despite this, the model's overall accuracy in its first month was 87% when predicting whether a given request would reach the required acceptance threshold (binary classification between acceptance and non-acceptance). This level of accuracy indicates that the model is already performing well, though there is room for further optimization.

Compared to earlier methodologies, the model has already enabled us to raise the cap from 6.5% to 7.5% of the total workforce per shift, a crucial step in increasing workforce flexibility while maintaining a reliable acceptance rate.

The current strategy for deciding the number of VETs to request follows a structured approach designed to maximize operational efficiency. If the number suggested by the model exceeds 7.5% of the total workforce scheduled for the shift, we adhere strictly to

the model's prediction. However, if the suggested number falls below the 7.5% threshold, Delivery Stations are given greater flexibility to request added VETs at their discretion, based on real-time operational needs. This hybrid approach ensures that workforce availability is still aligned with predicted demand while still allowing for necessary adjustments at the local level. The model's predictions suggest that this cap could be further increased, but we are approaching this change with caution, carefully analyzing the data before making further adjustments.

Thanks to the first wave of feedback, we are working on optimizing the model to provide a recommended range of VET requests rather than a fixed number. This will allow greater adaptability to different scenarios and offer Program Managers in the warehouses more flexibility in decision-making. By moving toward a range-based approach, we aim to reduce underutilization of available workforce capacity while still ensuring that acceptance rates remain at a high level. The next phase of the project will focus on refining these recommendations to make the model even more responsive to real-world conditions, ultimately improving both labor allocation efficiency and operational performance.

5.2 **Conclusions**

The introduction of the VET forecasting model marks a significant step toward a more structured and data-driven approach to workforce planning. While the model has already contributed to improving VET requests, its conservative nature has highlighted opportunities for improvement. By refining its ability to balance predicted acceptance rates with operational needs, we aim to enhance its effectiveness in aligning workforce allocation with real demand.

This project has already proved the potential of predictive modeling in workforce management, but its full impact will become clearer as we continue fine-tuning the system based on real-world feedback. The goal remains to strike the right balance between automation and operational decision-making, ensuring a more efficient and scalable VET management strategy.

BIBLIOGRAPHY

Amazon. (2024). *AMZL Internal Wiki*.

Ashish, R. (2023). *GeeksforGeeks*. Tratto da GeeksforGeeks.

Cortes, C., Mohri, M., & Rostamizadeh, A. (2012). *L2 Regularization for Learning Kernels.* arXiv preprint.

Guestrin, C., & Chen, T. (2016). XGBoost: A scalable tree boosting system. *International Conference on Knowledge Discovery and Data Mining (KDD '16)*, 785–794.

Hadavand-Siri, M., & Deutsch, C. (2012). Some Thoughts on Understanding Correlation Matrices. *CCG Annual Report*, p. 408-410.

Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied Logistic Regression (3rd ed.).* Wiley.

OpenAI. (2024). *Chat GPT*. Tratto da Chat GPT.

Permanasari, A., Hidayah, I., & Bustoni, I. (2013). SARIMA (Seasonal ARIMA) Implementation on Time Series to Forecast the Number of Malaria Incidence. *ICITEE 2013 Proceedings*, pp. 147-152.

Schapire, R., & Freund, Y. (1997). *A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting.*

Sonali, B., & Prinkaya, W. (2014). Research Paper on Basic of Artificial Neural Network. *IJRITCC*, 96-100.

Suresh Kumar, a. U. (2020). *Hands-on Exploratory Data Analysis with Python.* Birmingham: Packt Publishing.

INDEX OF FIGURES