Using Machine Learning for Robotic Answering of Taxi Request Phone Calls

Rafael Marques Ribeiro
Master in Computer Science
Department of Computer Science
Faculty of Sciences of the University of Porto
2024

Using Machine Learning for Robotic Answering of Taxi Request Phone Calls

Rafael Marques Ribeiro
Master in Computer Science
Department of Computer Science
2024

Supervisor
Michel Ferreira, Professor Auxiliar, Universidade do Porto

Co-supervisor
Rita P. Ribeiro, Professor Auxiliar, Universidade do Porto

“ On parle toujours mal quand on n’a rien à dire.”

Voltaire

Acknowledgements

I would like to express my gratitude to everyone who has supported me throughout this journey.

First, I would like to thank my supervisors, Prof. Michel and Prof. Rita P. Ribeiro, for their guidance and support. And to all the staff at Geolink, who were instrumental in the completion of this dissertation.

I would like to thank my university colleagues whom I have met over the years. A special thanks to my colleagues for always supporting me and always being willing to help,

Finally, I would like to thank my family for making this moment possible. To my mother Fernanda and my brother Micael, who have always been my biggest supporters and who have given up many things to pursue my studies.

Resumo

Hoje em dia, a procura de táxis continua a ser elevada. Uma das formas de os clientes pedirem um táxi é ligando para um *call center* de táxis. Uma vez que a maioria dos pedidos dos clientes segue determinados padrões, como fornecer a mesma morada de recolha que no pedido anterior, é possível melhorar a experiência do cliente prevendo a próxima morada de recolha para um pedido de táxi. Esta dissertação introduz uma abordagem que utiliza algoritmos de *machine learning* num ambiente de aprendizagem online, seguindo a metodologia CRISP-DM. A abordagem apresenta uma pipeline de duas etapas: a primeira etapa utiliza *clustering* para simplificar potenciais localizações e a segunda etapa envolve a classificação para prever o endereço de recolha. Um estudo experimental comparou múltiplos algoritmos e estratégias de aprendizagem. Os resultados obtidos neste estudo demonstraram que a abordagem proposta supera significativamente os métodos estatísticos simples na previsão do próximo endereço de recolha do cliente. Estes resultados indicaram que, se integrada num sistema de resposta interactiva de voz, a abordagem poderia melhorar a experiência do cliente. A solução foi efetivamente aplicada num cenário real, obtendo um feedback positivo dos clientes.

Palavras-chave: call center para táxis, previsão de endereços de recolha, serviço em tempo real serviço, resposta interactiva de voz (IVR)

Abstract

Nowadays, the demand for taxis continues to remain high. One of the ways customers can request a taxi is by calling a taxi call center. Since most customer requests follow certain patterns, like providing the same pickup address as in their previous request, it's possible to improve the customer experience by predicting the next pickup address for a taxi request. This dissertation introduces an approach that uses machine learning algorithms in an online learning setting, following the CRISP-DM methodology. The approach involves a two-step pipeline: the first step uses clustering to simplify potential locations, and the second step involves classification to predict the pickup address. An experimental study comparing multiple algorithms and learning strategies showed that the proposed approach significantly outperforms baseline statistical methods in predicting the customer's next pickup address. These findings indicated that if integrated into an Interactive Voice Response system, the approach could potentially enhance the customer experience. The solution was effectively applied in a real-life scenario, yielding positive customer feedback.

Keywords: taxi call center, pickup address forecasting, real-time service, Interactive Voice Response (IVR)

Table of Contents

List of Tables

List of Figures

List of Abbreviations

IVR Interactive Voice Response

ML Machine Learning

DBSCAN Density-Based Spatial Clustering of Applications with Noise

DT Decision Trees

KNN k-Nearest Neighbor

RF Random Forest

DTMF Dual Tone Multi-Frequency

CRISP-DM Cross-Industry Standard Process for Data Mining

AGI Asterisk Gateway Interface

CFE Circular Fuzzy Embedding

GPS Global Positioning System

API Application Programming Interface

AI Artificial Intelligence

1. Introduction

This chapter provides an introduction to the motivation and goals of this dissertation in the context of a particular business scenario. Specifically, it outlines the objective of enhancing the taxi ordering process using machine learning integrated into an IVR system. Finally, it provides an outline of the dissertation's structure.

1.1. Motivation

Geolink is one of the largest taxi software companies in Portugal and one of the services available is to request a taxi through its call center, which receives thousands of taxi requests by phone every day from all over the country. Despite the high volume of calls, many customers provide the same addresses as their pickup location, indicating a pattern of service use. In this context, we believe that the process of ordering a taxi by phone can be optimized by using each client's call history and by implementing an Interactive Voice Response (IVR) that suggests the foreseen address.

An IVR, named Napoleão, is already operational in the company's call center for specific clients that only request one pickup address, intending to measure the reception of customers who use this type of service. Using a human-like voice, generated by text-to-speech using Wavenet [23], provided by Google Cloud [10], the customer is served by the robot, which suggests the most frequent pickup address, after which the customer has to validate the suggestion by Dual Tone Multi-Frequency (DTMF) keypad selection to request a taxi or pass it on to the human operator. In May 2024, 1 in 5 calls were answered by Napoleão.

1.2. Objectives

The ultimate goal is to implement this solution in an IVR [16] system which is connected to the call center so that customer calls are answered by an automatic voice that suggests a pickup place according to the location predicted by the machine learning model. By

analyzing each customer's call history, we aim to develop a solution to predict the pickup address and suggest it to the customer when they call to make a taxi order.

We start by conducting a study to analyze the best solution to implement in the IVR. From this study, we attained an online Machine Learning (ML) approach consisting of two steps to achieve our goal. The first step used clustering to group pickup points that were geographically similar. In the second step, a classification was performed to forecast the next pickup address based on the customer's call history and clusters.

Figure 1.1 represents the general architecture using the machine learning approach.

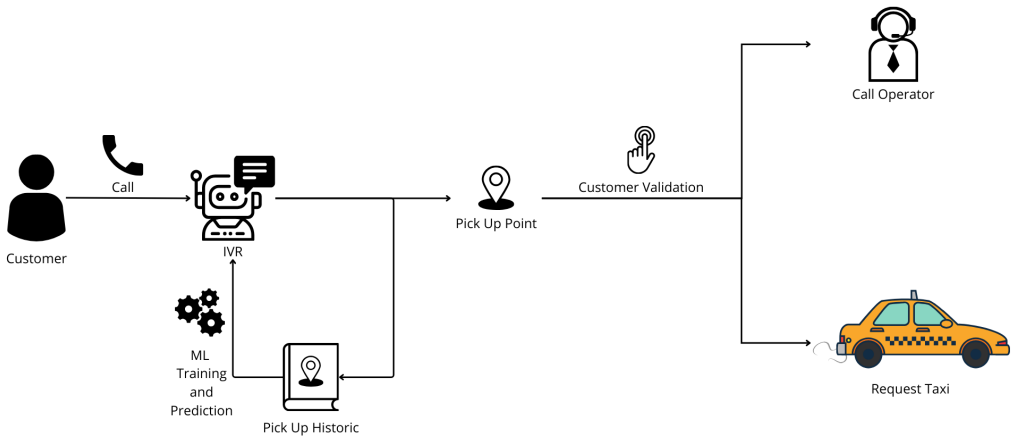


Figure 1.1: Architecture for automating taxi request phone call answering using machine learning.

To evaluate the effectiveness of the approach, we conducted tests using different clustering and classification algorithms and online learning setups on a real-world dataset. Additionally, the approach was compared to baseline methods, such as recommending the most frequent address or using the last pickup address. The results obtained over a set of customers indicate a significant improvement in predicting customers' pickup locations.

Overall, this approach has the potential to enhance the efficiency of the taxi ordering process while also providing a better customer experience. Machine learning can significantly optimize a call center's IVR processes. A positive experience helps maintain customer satisfaction and attract new ones. Additionally, increasing the number of taxi requests handled by an automated IVR operator reduces the economic cost of the call center, making it more competitive with other taxi-ordering methods like smartphone applications.

1.3. Document Structure

This document is structured as follows. Chapter 2 reviews the topics and the related work on using machine learning to enhance taxi services, call centers and IVR. Chapter 3 describes the adopted methodology and covers the initial stages (data set, pre-processing methods and introduction to ML approach and the algorithms used). In Chapter 4, the possible approaches were tried and evaluated. Chapter 5 describes the implementation of the ML approach in the IVR system. Chapter 6 concludes this dissertation.

2. Background

This chapter introduces the Interactive Voice Response (IVR) system and basic learning tasks in Machine Learning (ML). It also provides an overview of related literature and concludes with a summary contextualizing this study.

2.1. Interactive Voice Response

A call center is a centralized office or facility that receives or makes a large volume of telephone calls, typically for customer service, support, or telemarketing. Call centers aim to enhance customer satisfaction and streamline business communication. Utilizing technology for call routing, monitoring, and analytics improves efficiency and service quality.

An Interactive Voice Response system is an automated telephony technology that allows callers to interact with a computer-operated phone system through voice commands or touch-tone keypad inputs. IVR systems are commonly used in call centers and customer service departments to route calls, provide information, and perform basic transactions. These systems help businesses handle high call volumes efficiently, reduce operational costs, and provide 24/7 service availability, enhancing the overall customer experience.

2.2. Machine Learning

Machine Learning has seen an explosion in recent years, where advancements in algorithms, computing power, and data availability have driven unprecedented progress. This growth has enabled significant breakthroughs across various fields, such as healthcare, finance, and autonomous systems. Call centers are evolving thanks to ML, making them more autonomous through the use of IVR.

In the context of the data, the ML approach chosen was a combination of unsupervised and supervised algorithms.

Unsupervised learning algorithms are designed to identify patterns in data without the need for labeled outcomes, often used for clustering, association, and dimensionality reduction. Two notable examples are K-Means and DBSCAN. K-Means [18] is a clustering algorithm that partitions data into K distinct clusters based on feature similarity. It iteratively assigns data points to clusters and updates the cluster centroids. DBSCAN [8], on the other hand, groups together points that are closely packed and marks points in low-density regions as outliers.

Supervised learning algorithms predict outcomes based on labeled training data and are commonly applied to classification and regression tasks. Noteworthy examples include k-Nearest Neighbor (KNN) [12], Decision Trees (DT) [13], and Random Forest (RF) [4]. KNN is a classification algorithm that assigns a data point to the most common class among its K nearest neighbours in the feature space, known for its simplicity and effectiveness in many problems. DT are models that make decisions based on a tree-like structure of decisions. RF, an ensemble learning method, constructs multiple decision trees during training and outputs the mode of the classes (for classification) of the individual trees.

2.3. Related Work

This section reviews studies that relate to ML applications for call centers and IVR systems. Through the analysis of various methodologies and algorithms, this review looks at the advances in the use of ML to optimize customer interactions.

2.3.1 Trip Prediction

Several research works have focused on improving taxi operations using ML. For instance, predicting taxi-passenger demand at each taxi stand to recommend the next stop for a taxi is described in [21].

Taxi trip destination prediction is also a commonly studied topic in the machine learning community. Typically, the main focus of these studies is to optimize taxi locations for efficient customer service and savings for the taxi service provider. In this regard, we highlight some of the studies below.

Eberstein et al. [7] developed an online framework for trip destination prediction based on the methodology suggested by Ashbrook and Starner [2]. This methodology used

geographical coordinates to find locations via the k-Means [18] clustering algorithm. The authors implemented an online learning strategy that updates the model according to the customer's history for new call arrivals.

In [1], the authors conducted a study to predict trip destinations by combining the Markov Chain and Multinomial Logit model to create patterns using the travel traject order. This proposal relied on a previous study [28] that suggested using the Hidden Markov Model from the past Global Positioning System (GPS) log and the current location to predict a user's destination when beginning a new trip.

In addition to taxi destination prediction, other studies have addressed related problems. Zong et al. [28] designed an ensemble approach that predicts a taxi's remaining journey time. In a recent work by Liao et al. [17], a taxi destination prediction was performed using ensemble learning and Circular Fuzzy Embedding (CFE) to represent time-related features.

2.3.2 Call Center IVR

This literature review explores various facets of call center and IVR technologies, examining their impact on customer satisfaction, operational costs, and service delivery across different industries.

Agent scheduling in call centers is a significant management challenge due to the difficulty in balancing service quality with costs. Traditional methods like regression and time series analysis predict future call arrival counts. [22] introduces a novel approach by discretizing these counts into finite intervals to identify better demand peaks that cause abandoned calls. This is achieved using multi-class classification. The proposed method was tested on real-world data from a taxi dispatching call center. Results show that this approach effectively reduces the number of abandoned calls while keeping staffing costs reasonable. In the article by Mehrbod et al. [20] emphasize the critical role of call centers in customer interactions and the importance of optimizing their performance for both cost reduction and improved customer satisfaction. The authors propose using machine learning techniques to enhance call center operations by predicting the outcome of pairing callers with agents based on historical and demographic data. This approach aims to intelligently match callers with the most suitable agents, leveraging past data to achieve

better performance metrics. The results indicate that such predictive models offer reasonable performance improvements, to refine and enhance functionality in the future.

More recently, a study by Filippou et al. [9] aimed to minimize the waiting time of a customer through call center IVR using ML algorithms.

2.4. Summary

Throughout this chapter, various works related to the theme have been introduced and carefully analyzed. Based on this analysis, a two-step machine learning approach was determined to be the most effective strategy. However, as far as we know, no prior research has explored predicting a customer's next pickup address. This project investigates different clustering and classification algorithms using various online learning setups on the customer's call history data, drawing upon existing literature for related tasks.

3. Methodology

This chapter presents the methodology followed throughout the work and briefly describes it. After that, the data manipulation steps are unfolded in detail.

3.1. CRISP-DM Process

Choosing an appropriate methodology is key to ensuring a structured and efficient approach to extracting meaningful insights from datasets.

For this study, was adopted the Cross-Industry Standard Process for Data Mining (CRISP-DM) [27] as the guiding framework. Being the most common methodology for data science projects, CRISP-DM is widely recognized and utilized, particularly in business contexts.

Each step in the CRISP-DM process is iterative, meaning that insights or issues discovered at later phases can lead to revisiting earlier phases for refinement. This flexibility helps ensure the final model is as accurate and useful as possible. CRISP-DM is defined in six phases, as depicted in Figure 3.1.

The process starts with understanding the project objectives and requirements from a business perspective. Next, the data understanding phase involves collecting initial data to become familiar with it and uncover initial insights. This step is crucial for identifying the characteristics and quality of the data. The data preparation phase follows, where the final dataset is prepared for modeling. This step includes cleaning, transforming, and organizing the data to ensure it is suitable for analysis. Once the data is ready, the modeling phase involves applying various techniques and calibrating their parameters to optimal values. This process involves experimenting with different algorithms and configurations to find the best-performing model. After modeling, the evaluation phase assesses the model's performance and ensures it meets the business objectives. Finally, the deployment phase involves deploying the model into the operational environment, making it accessible and usable for practical applications. This phase ensures that the model can deliver value in a real-world setting.

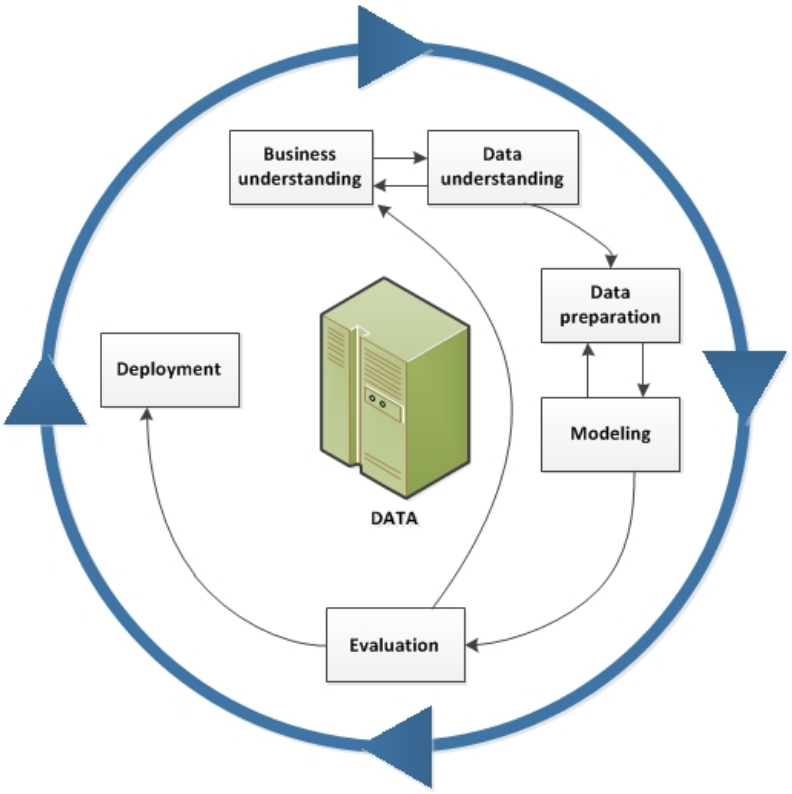


Figure 3.1: CRISP-DM Diagram [14].

The following sections and chapters will provide a detailed analysis of each phase of the CRISP-DM methodology. This comprehensive analysis will include the specific activities, the methods and tools employed, and the rationale behind key decisions.

3.2. Business Understanding

The project aims to implement a solution in an IVR system for the Geolink call center, allowing an automated voice to suggest a pickup location based on an ML model. By analyzing each client's call history, the solution predicts and suggests the pickup address when the customer calls to order a taxi.

3.3. Data Understanding

The dataset used in this study is a compilation of taxi orders placed by customers to a taxi call center in the Lisbon district. It consists of 107252 calls made by 7651 customers from February to November 2023. The data comprises the customer's phone number, the order date and time, and the pickup address name. Figure 3.2 shows the percentage

of different pickup addresses requested by customers. As can be observed, the majority of customers use the same address.

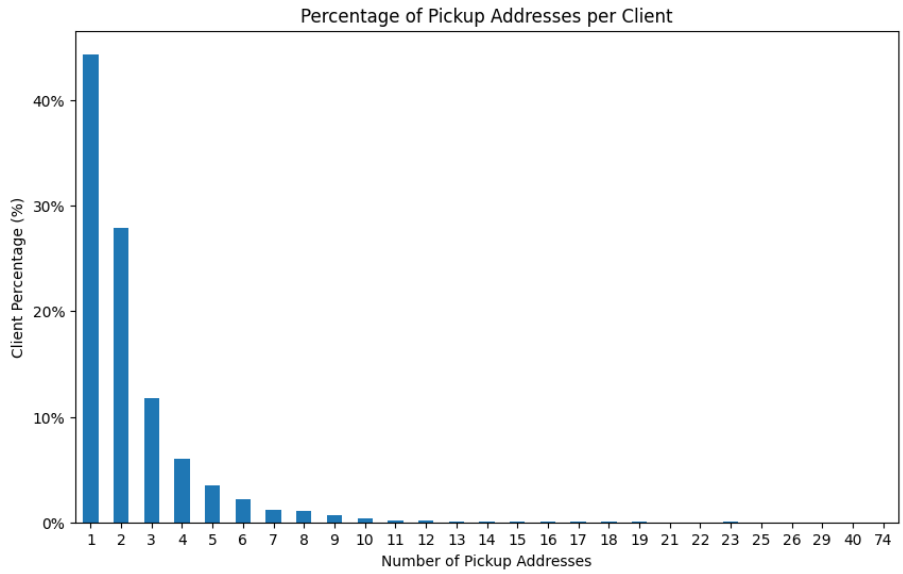


Figure 3.2: Distribution of the percentage of pickup addresses per customer.

Figure 3.3 shows the number of calls made during different times of the day. It is noticeable that the majority of calls are made in the morning. However, on weekends, there is an increase in calls during nighttime and early morning.

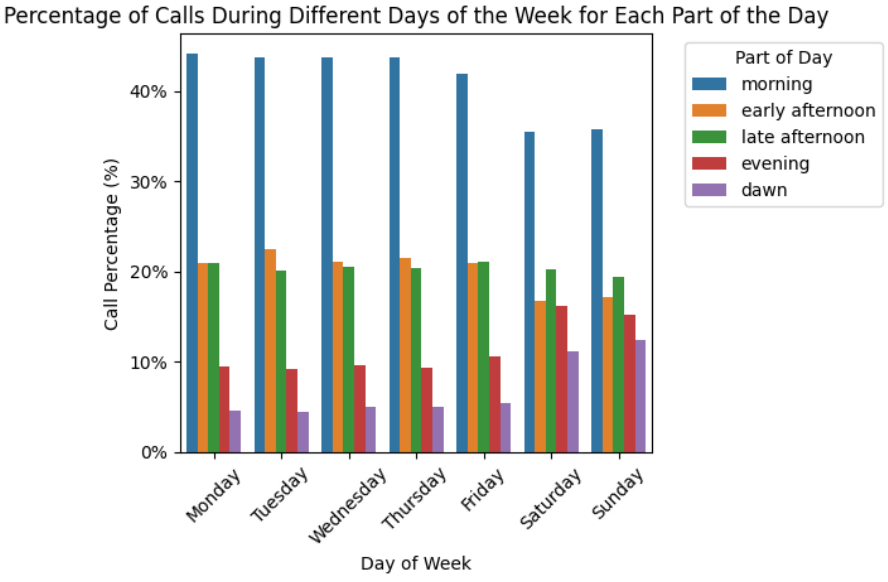


Figure 3.3: Distribution of phone calls percentage per weekday and part of the day.

Figure 3.4 shows that cell phones are the most popular device used by customers, with around 80% of customers using them. Afterwards, we have telephones, with only 20% of customers using them, and it can also be deduced that these customers always order their orders in the same place, since that's where the call is made.

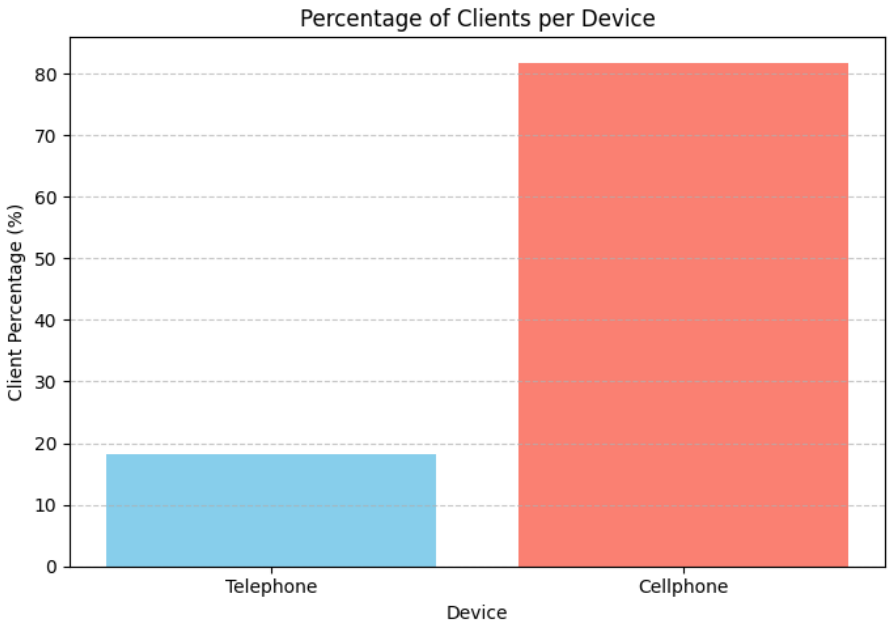


Figure 3.4: Client distribution by device.

3.4. Data Preparation

After conducting the preliminary exploratory data analysis, we performed feature engineering to incorporate time and spatial information. This will enable us to apply classical clustering and classification algorithms.

The original data set was partitioned into multiple data sets, one for each customer's phone number. In this dataset, the minimum number of calls is 2, and the maximum is 857. On average, customers make 12 calls. Additionally, new features were created to incorporate information related to the taxi order date and time as well as the pickup address.

Weekday and Weekend. The date of each phone call was transformed into a boolean attribute `weekday`, indicating whether it is a weekday (1) or a weekend (0). For now, public holidays have not been considered in this study.

Period of the Day. The time of each phone call was assigned to a period of the day by applying the data embedding technique referred to in [17]. The day was divided into five periods: Morning (between 6h-13h), Early Afternoon (between 13h-16h), Late Afternoon (between 16h-20h), Evening (between 20h-24h), and Dawn (24h-6h). The afternoon was divided into early and late because it was a long period with typically too many orders. Two

new attributes were created using trigonometric expressions based on the original time of the call to capture the cyclical nature of these periods, as presented in Equations 3.1 and 3.2.

$$\texttt{dayPeriod\_sin} = \sin\left(2\pi p/n\right) \tag{3.1}$$

$$\texttt{dayPeriod\_cos} = \cos\left(2\pi p/n\right) \tag{3.2}$$

, where p represents the period of the day (Morning-1, Early Afternoon-2, Late Afternoon-3, Evening-4, Dawn-5) and n, is the number of periods considered within the day, which in this case is 5. Figure 3.5 plots the considered five periods of the day according to the trigonometric encoding, using sine and cosine.

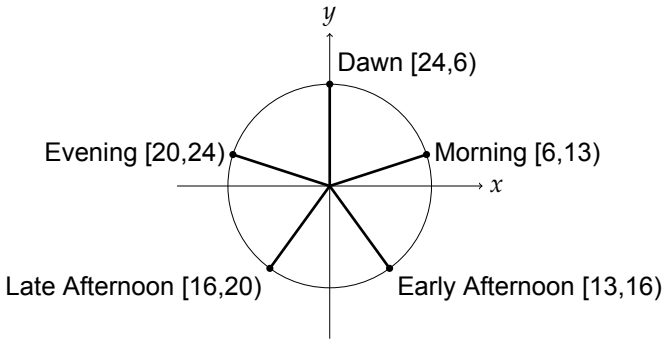


Figure 3.5: Cyclical embedding of the time of a phone call into a period of the day.

GPS Coordinates. The geographical coordinates (latitude and longitude) for each pickup address were obtained, the values used are the Geolink reference values for each address. Thus, two new features were created, `pickup_lat` and `pickup_lon`, which describe the precise location of the pickup. These new features replaced the previous pickup address. This transformation is essential as using the original street name can sometimes be inaccurate due to possible abbreviations or multiple names for the same location, which could lead to errors later in the pickup address prediction phase.

3.5. Modeling

The approach for addressing the problem at hand involves using machine learning, as outlined in the diagram in Figure 3.6. The first step is to perform a clustering task to group pickup points based on their geographical location. This step serves as a filter to ensure precise locations defined as classes. Once this is done, the classification task predicts

the class that represents the next pickup location based on the attributes of the phone call.

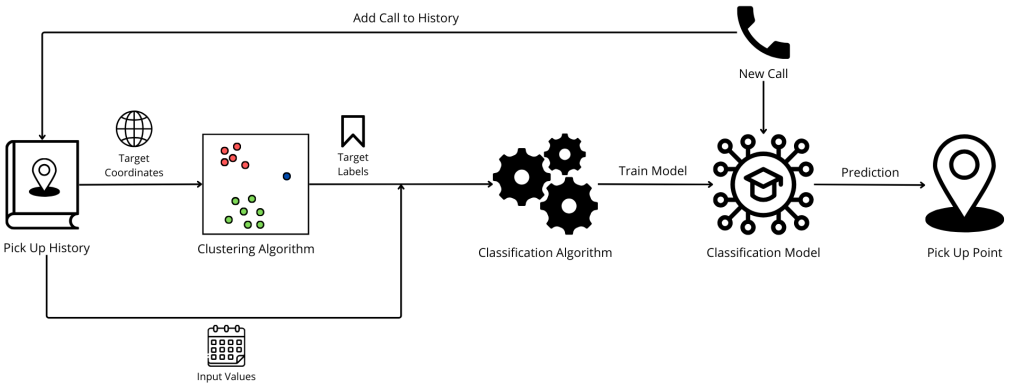


Figure 3.6: Machine learning pipeline to predict the pickup point for a phone call.

3.5.1 Clustering Task

The clustering is based on latitude and longitude coordinates, as some customers may have the same street or door but are written differently in the dataset, and the coordinates may not be the same, but approximate, since the correct name of the address is not known. This helps the training phase of the classification algorithm by reducing the number of distinct classes.

This study examined two clustering approaches: one based on distance - k-Means [18], and the other based on local density - DBSCAN [8].

The *Haversine* distance [25] was used as the distance metric for both clustering algorithms. This metric provides a good approximation for the distance between two points on the Earth's surface. It has an average error rate of less than 1%, as reported in [24]. This is particularly useful for clustering geographical coordinates represented by latitude and longitude.

Thus, given a geolocation $x = (x_{\text{lat}}, x_{\text{lon}})$ and $y = (y_{\text{lat}}, y_{\text{lon}})$, the *Haversine* distance $D(x, y)$ is defined by the Equation 3.3 and depicted in Figure 3.7.

$$D(x,y) = 2\arcsin\left[\sqrt{\sin^2\left(\frac{x_{\text{lat}} - y_{\text{lat}}}{2}\right) + \cos(x_{\text{lat}})\cos(y_{\text{lat}})\sin^2\left(\frac{x_{\text{lon}} - y_{\text{lon}}}{2}\right)}\right] \tag{3.3}$$

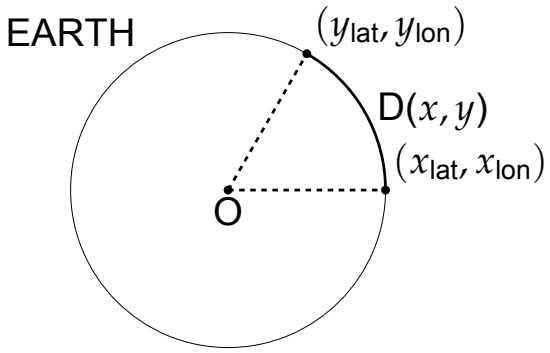


Figure 3.7: *Haversine* distance $D(x, y)$ between two geolocated points x and y.

For illustration purposes, Figure 3.8 shows the result of the DBSCAN clustering algorithm for a particular customer, which had only five different addresses in its history, each forming a cluster.

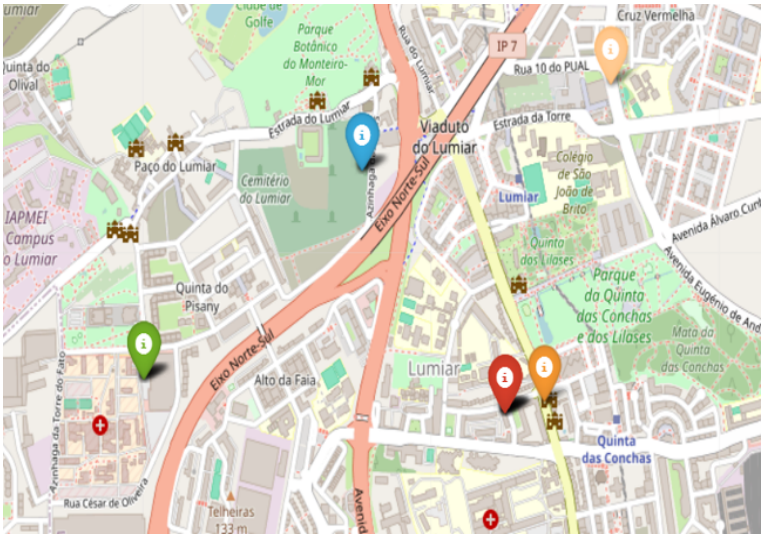


Figure 3.8: Example of customer pickup points after clustering with DBSCAN.

3.5.2 Classification Task

The next task focuses on classification, wherein a model is trained to predict the pickup location (class) identified by the cluster formed in the preceding task. This prediction relies on the weekday and time of day of the call.

The chosen classification algorithms were: KNN [12] for its simplicity, CART DT [13] for its effectiveness and interpretability, and RF [4] for its competitive effectiveness, leveraging decision trees.

After predicting the location or class, an address needs to be provided for the IVR to suggest to the customer. This is achieved by selecting the most frequent address within the cluster.

4. Evaluation

This chapter discusses the experiments conducted to determine if the proposed ML approach is superior to simple strategies like predicting the most frequent or last pickup address in the customer's history. This chapter explains the experimental setup and presents and discusses the obtained results. All machine learning algorithms used in this study were implemented using scikit-learn [24].

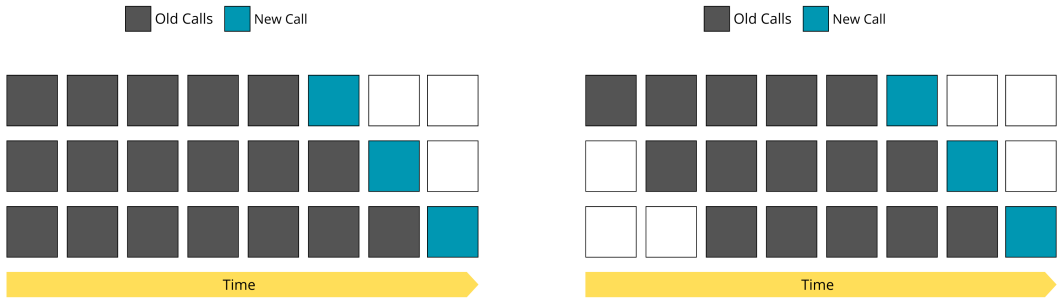
4.1. Experimental Setup

In our study, we trained a separate prediction model for each customer's next pickup location. However, some customers make very few calls or only request pickups from the same location, so using a machine learning model is unnecessary. Therefore, for this experimental study, it was decided that only customers who have made at least 5 or 10 phone calls and have requested pickup addresses from two different locations will be included in the machine learning approach. For customers who do not meet these conditions, their intended address will be determined based on the most frequent or the last one requested. In the dataset, when the minimum number of calls is set to 5, the machine learning approach uses 31% of the customers' call histories. However, if the minimum is set to 10, the percentage of customers included in the machine learning approach decreases to 19%.

The experiments used different online learning and algorithm setups, as described next.

Online Learning Setup To better capture evolving changes in the usage patterns of the phone-requested taxi service, two scenarios were considered for online learning: a growing window and a sliding window, depicted in Figures 4.1a and 4.1b, respectively, and detailed next.

- Growing Window: in this setup, the model is trained using all the calls available in the history. The model is first used to make a prediction whenever a new call arrives, and then the new call is added to the history.
- Sliding Window: the model is trained using the minimum number of calls as the training data in this setup. Every time a new call arrives, a prediction is made, and the oldest call in the batch is removed from the history. The latest call is added to the training data, and the model is trained again with the updated data, This strategy forces the models to focus on the most recent history, potentially allowing them to capture changes better.



(a) growing window starting with 5 calls (b) sliding window with the last 5 calls

Figure 4.1: Online learning setups per customer.

Growing and sliding window setups were tested with a minimum threshold of 5 or 10 calls.

Algorithms Setup Due to the small number of examples available to train the models, which in the case of the sliding window is 5 or 10, extensive hyper-parameter tuning was not performed at this stage. As such, the hyperparameter values, shown in Table 4.1, were set as stated below.

For the k-Means algorithm, the optimal number of clusters for the customer's calls is determined between 1 and the maximum number of clusters based on the number of different addresses. The silhouette score [26] was calculated to find the best number of points that provide the ideal number of clusters to create. For the DBSCAN algorithm, the eps parameter is calculated by dividing the distance of 250 meters by the equatorial radius of the Earth (6371000 meters, according to [19]). This conversion transforms the distance into a suitable eps value that reflects the relative scale of 250 meters compared to the Earth's size. This distance was chosen to prevent clustering points that are too close together and might represent the same location, such as the same avenue or street.

In the classification step, the number of neighbors for KNN is determined using the minimum number of addresses as a threshold. If the minimum value is 5, then the number of neighbours is set to 3; if it is 10, then the number of neighbours is set to 7. A large tree was allowed to grow without constraints for the CART DT algorithm. As for the RF algorithm, 50 trees were used without pruning.

Table 4.1: Hyperparameters set for clustering and classification algorithms.

Algorithm	Hyperparameters
k-Means	n_clusters={1:max_address}
DBSCAN	eps=250/6371000, min_samples=1
knn	n_neighbors={3,7}, metric='minkowski', p=2
dt	max_depth=None, min_samples_split=2, min_samples_leaf =1
rf	max_depth=None, min_samples_split=2, min_samples_leaf =1 max_leaf_nodes=None, n_estimators=50

Evaluation Setup The performance of different prediction methods was evaluated using accuracy as the evaluation metric. A prequential approach was employed to assess the model's accuracy over time. To estimate the performance of each prediction method, the average and standard deviation of the accuracy were computed from all predicted addresses for each customer.

Since each customer constitutes a different dataset, a multi-comparison statistical testing procedure was conducted, following the approach suggested by Demsar [6], to assess the significance of variation in performance estimates obtained by the prediction methods across multiple datasets. The implementation provided in [5] was utilized. The Friedman test was employed as the omnibus test, and the Holm test was the post-hoc test. To identify statistically significant differences at a 95% confidence level, the p-value was set to 0.05.

4.2. Results

Table 4.2 shows the results of the experimental study, using all customers. As mentioned earlier, the prediction is made using a basic statistic when the history has less than the minimum number of orders. Here, the decision was made to use the most frequent address since it demonstrated higher accuracy compared to using the last address. From the results, it is evident that these non-ML approaches on their own have the lowest accuracy compared to the other tested methods. Overall, combining k-means clustering with

one of the classification algorithms produces the best results. This is especially meaningful when considering that the model is trained using a sliding window of the last 5 calls. Conversely, when using the DBSCAN clustering, the best results are obtained using a growing window with 5 calls in the history as a starting point. As for the classification algorithms, the differences are not always consistent regarding the best performance.

Table 4.2: Mean and standard deviation of prequential accuracy for 7651 customers from February to November of 2023, using the two-step approach and the baseline prediction methods of last or most frequent (freq) pickup address. The best results achieved by each variant of prediction method are highlighted in bold.

Prediction		Growing Window (gw)		Sliding Window (sw)	
Method		init=5	init=10	size=5	size=10
kmeans	knn	0.729 ± 0.379	0.700 ± 0.382	**0.741 ± 0.378**	0.699 ± 0.384
	dt	0.732 ± 0.378	0.695 ± 0.386	**0.742 ± 0.378**	0.698 ± 0.383
	rf	0.732 ± 0.377	0.696 ± 0.383	**0.740 ± 0.379**	0.704 ± 0.382
dbscan	knn	**0.706 ± 0.388**	0.700 ± 0.383	0.704 ± 0.388	0.699 ± 0.382
	dt	**0.707 ± 0.385**	0.695 ± 0.376	0.703 ± 0.386	0.697 ± 0.382
	rf	**0.708 ± 0.385**	0.698 ± 0.382	0.705 ± 0.386	0.698 ± 0.382
freq		**0.686 ± 0.377**			
last		**0.661 ± 0.384**			

Figure 4.2 presents the ranking of various methods and their respective critical differences that help to determine the significance of the differences among them. All methods tested are presented in Appendix A

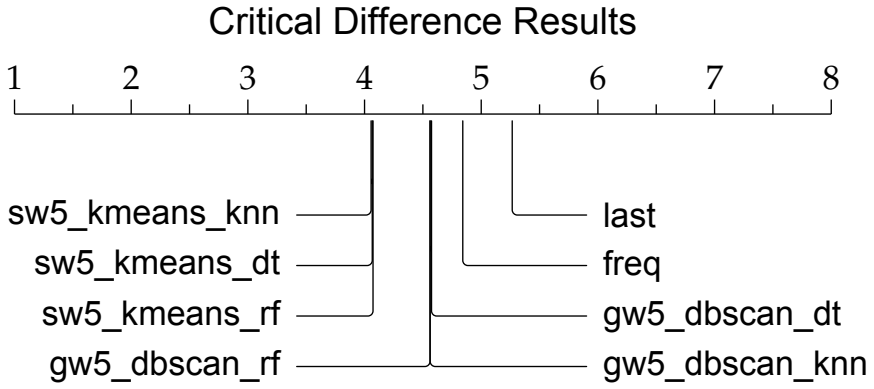


Figure 4.2: Critical difference diagram for 95% confidence level for the best variants of each tested prediction method.

As can be observed, the approach that implements a sliding window of size 5 with k-Means obtains significantly better results than the other methods.

The k-Means and classification algorithms performed better than DBSCAN. This may be due to the search for the k optimal clusters calculation that makes the algorithm more precise. However, k-means had some cases where points within a few meters of each other were clustered together, even if the addresses were different. The DBSCAN algorithm avoids this by setting an appropriate value of eps. Though this may lead to more classes to predict, the clusters look more precise than the ones created by k-Means.

When evaluating the clustering algorithm's results for each online learning setup, it was found that DBSCAN performed better with the growing window. In contrast, k-Means performed better with the sliding window. This may be due to the fact that DBSCAN can create more meaningful clusters with large amounts of data.

5. Deployment

This chapter focuses on the culmination of the CRISP-DM process, where the final model is deployed into the operational system and integrated into the IVR system for real-world testing. The outcomes of these tests are detailed in the implementation test section of this chapter.

5.1. Create and Update Model

To create or update the ML model, a pipeline was developed to ensure the certification of this process using several technologies.

This implementation details a machine learning-enhanced call center system. By integrating various components and employing advanced algorithms. Figure 5.1 represents the architecture that creates the ML model per client and updates it every time a new request call is made.

The implementation for the machine learning application was built with Python. Here's a breakdown of the components:

- Call Center Software: It is responsible for managing incoming calls from clients, using the Asterisk Gateway Interface (AGI) [3] library, which can interact with the telephony hardware.
- Database: The call center software stores call data in a database. This could include information about the caller, the pickup localization, the destination localization, and the call duration.
- Kafka [15]: a publish-subscribe messaging system. The call center software publishes events (e.g., a new call is coming in) to a Kafka topic. Other applications can subscribe to the topic to receive these events.

- Machine Learning Application: The machine learning application subscribes to Kafka topics to get insights from call center data. For instance, it could be used to analyze call recordings to identify common customer issues or to improve the effectiveness of the call center's IVR system.
- IVR Application Programming Interface (API): Works like bridge that connects the database to the ML application and to the IVR system.

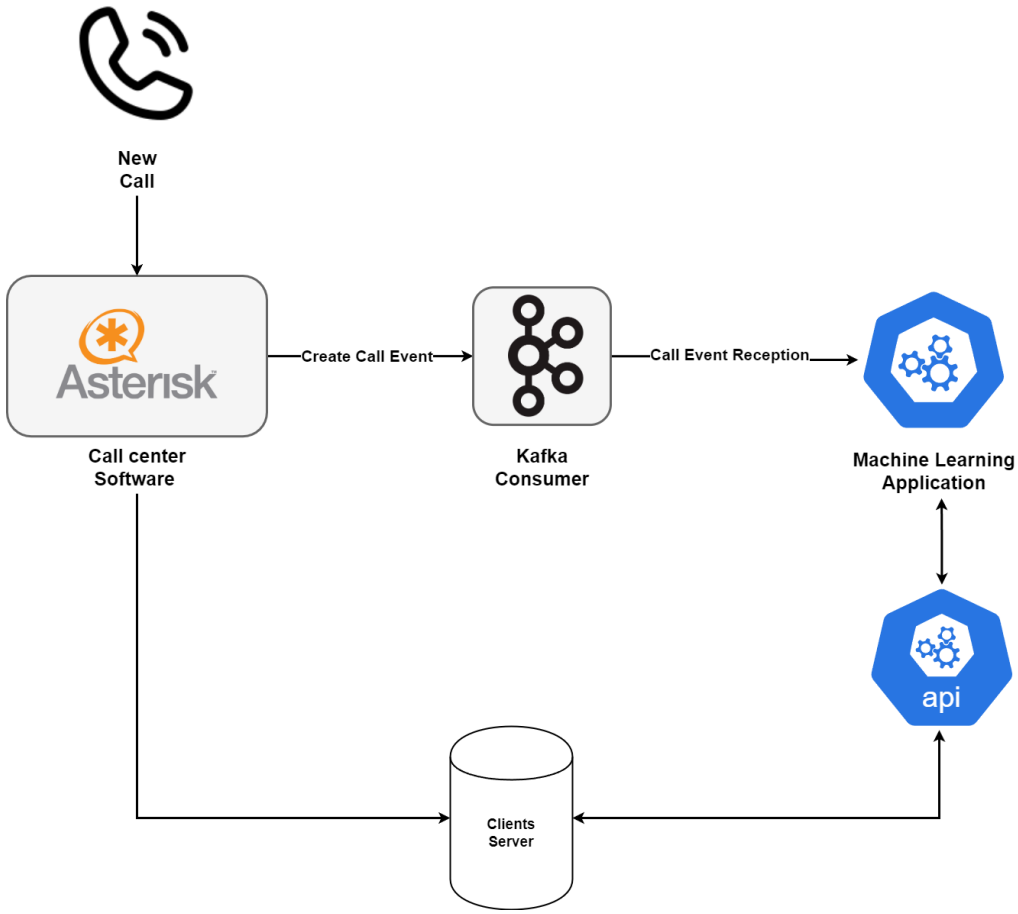


Figure 5.1: The embedding of machine learning in the system.

In general, this architecture allows for a modular and scalable call center system. Call center software can focus on its primary responsibility of managing calls, while other applications can consume call data to provide additional functionality.

5.1.1 Machine Learning Application

In the ML application, the customer's history is accessed using a growing window but using a maximum of the last 30 orders so that training can be carried out more quickly. With that information, a model is trained with DBSCAN (clustering) and DT (classification), and a binary file is created, which stores the model developed for the client. This file is stored in the database. Whenever a customer calls the call center, the file is called to predict the pickup address according to the current time of the request.

DBSCAN was selected for the clustering task over k-Means because it is more suitable for handling geographical coordinates. For the classification task, DT was chosen due to its simplicity and speed, offering more complexity compared to KNN.

5.1.2 API

An API was created on purpose for the implementation, and some endpoints were created to upload and download the binary files, collect the customer's history; and update the readable names that are used by IVR to suggest to the client the pickup address, for creating readable names, the Vertex AI [11] platform from Google Cloud API [10] is utilized, that uses generated Artificial Intelligence (AI); and other to predict the pickup location.

5.2. Operator Interface

The operator has a terminal at his disposal to dispatch the calls. Figure 5.2 shows a snippet of the terminal with a specific client using ML algorithms to predict the next pickup address. According to the ML algorithms, there are three candidate addresses for this example. In addition, you can manually edit the readable name that is spoken by IVR, in case the name generated is wrong or you need to add more information.

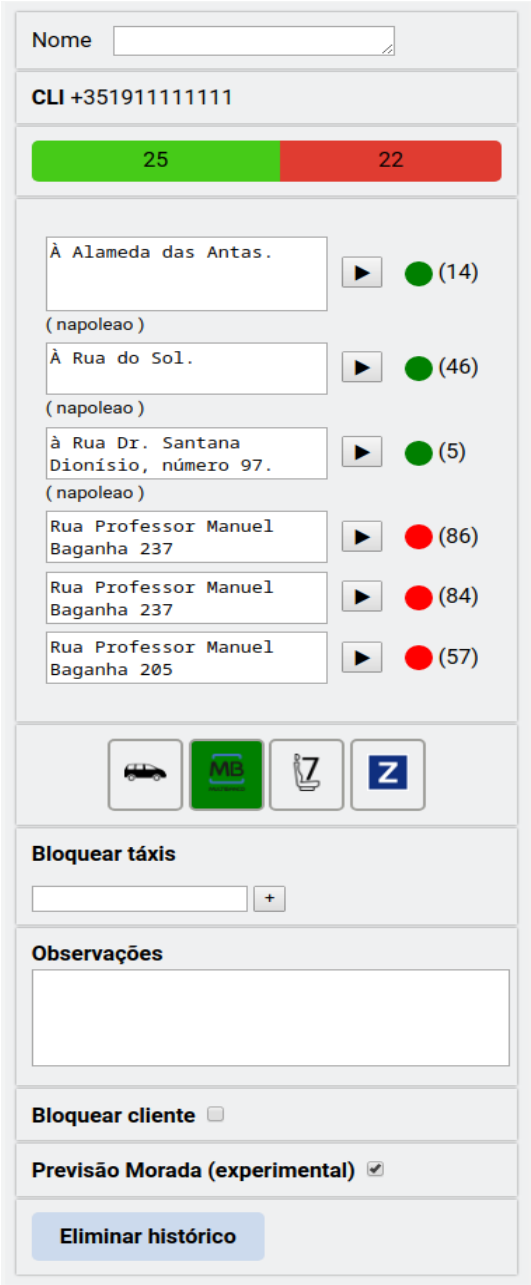


Figure 5.2: Snippet from the operator terminal.

5.3. Implementation Test

To test the solution in a real context, initially, 10 customers were carefully selected according to their history and activity. The experiment aimed to evaluate the effectiveness of using machine learning models for robotic answering of taxi request phone calls. Figure 5.3 shows the distribution of clients inserted into this test per Portugal's regions.

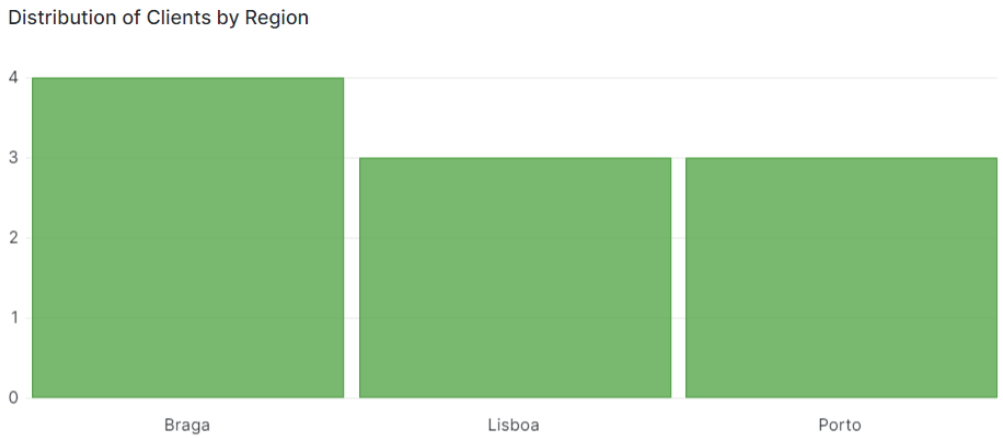


Figure 5.3: Client distribution using models to predict the pickup address per region.

These client numbers were selected mainly because they present a pattern that, will be learned by the models. This pattern follows the rule of not having too many calls to different addresses at the same period of the day.

Over the test period from June 19th to 26th, 67 calls were analyzed, with the system suggesting the pickup addresses. During the first days, additional numbers were incorporated as the experiment progressed.

Figure 5.4, displays the actions taken by the clients when the call is answered by the robot (Napoleão). Clients can take four actions: Napoleão, is when the client accepts to call a taxi to the predicted pickup address; Operator, is when the client doesn't accept the predicted address or when want to ask for any specification like the type of payment, route alternatives, etc; Disconnected, happens when the client hangs up the phone, while the robotic voice is speaking; And the last action is when the client doesn't press any button and the call is transferred to the operator.

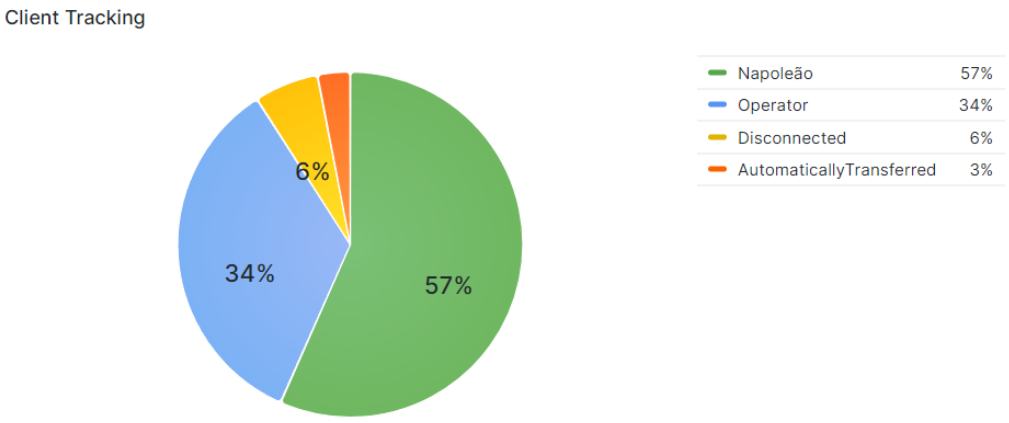


Figure 5.4: Clients call track actions while using machine learning models to predict the pickup address.

The results indicate a positive outcome, as the majority of calls (57%) were successfully handled by the robotic system (Napoleão) with clients accepting the suggested pickup addresses. A significant portion (34%) of calls required further assistance from a human operator, indicating that while the robotic system was effective, there was still a substantial need for human intervention.

Additionally, the data revealed a small percentage of calls that were disconnected (6%) or automatically transferred (3%). It should be noted that most of the customers who initially disconnected have called back, which suggests that a lack of familiarity with the automated system may have been a factor in their initial disconnection.

Overall, the high acceptance rate by Napoleão demonstrates the potential for machine learning models to efficiently handle taxi request calls, though further refinement and user familiarization may enhance overall performance and user satisfaction.

6. Conclusion

This final chapter concludes the dissertation. It synthesizes the findings and suggests potential directions for future work.

6.1. Contributions

We presented a two-step approach for predicting the next pickup address for a customer's taxi request phone call. In the first step, we employed clustering to merge the street names based on the geographical coordinates of previous customer's pickup locations. Afterwards, we resort to a classification model learnt from historical taxi request data to predict the pickup location for the customer's next taxi request.

Real-world data was used from phone calls spanning nine months in a specific district to test the approach. The overall evaluation of the performance of the approach, comparing the effectiveness of two online learning configurations with different clustering algorithms (k-Means and DBSCAN) and classification algorithms (KNN, CART DT and RF). Experiments showed that the proposed solution outperformed baseline solutions, such as predicting the most frequent or recent pickup address. Moreover, the sliding window strategy that uses only the last five phone calls achieved the best results. These preliminary results confirm the initial hypothesis that many customers exhibit a service use pattern and that it is fast-evolving in many cases.

In deployment phase, this approach obtained good results in the machine learning algorithms and customer acceptance, showing the potential for robotic systems in the taxi call center. The acceptance rate indicates that customers are receptive to automated solutions, especially when they are efficient. The need for human intervention in certain cases points to opportunities for improvement and refinement of the algorithms. With continued development, such automated systems can provide reliable and cost-effective solutions for managing customer service interactions.

The main objective of this dissertation was accomplished. This approach in a real-world setting showed promising results in both machine learning performance and customer acceptance, indicating the potential for integrating automated systems in taxi call centers.

6.2. Future Work

This was the first approach to the problem, and we found points for improvement that could be addressed in the future. In future work, the effectiveness of this approach will be studied further by analyzing a larger sample of data, including different regions of the country and longer time frames. One study could be the impact on the solution of splitting the morning since this is the period with the most requests. Additionally, in accordance with the CRISP-DM methodology, the process will be revisited and refined at each stage. As part of the research, ways of enabling the models to recognize holidays will also be considered, which will further increase the accuracy and relevance of the forecasts.

A. Experimental Results

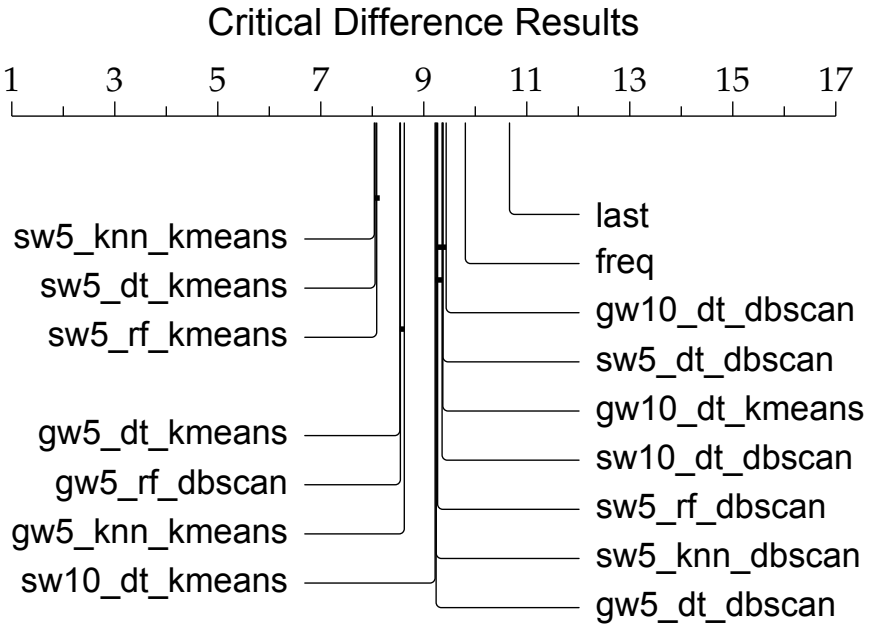


Figure A.1: Critical difference diagram for 95% confidence level for all variants tested as a prediction method.

References

[1] Alvarez-Garcia, J., Ortega, J., Gonzalez-Abril, L., Velasco, F.: Trip destination prediction based on past gps log using a hidden markov model. Expert Systems with Applications **37**(12), 8166–8171 (2010). https://doi.org/10.1016/j.eswa.2010.05.070 [Cited on page 7.]

[2] Ashbrook, D., Starner, T.: Starner, t.: Using gps to learn significant locations and predict movement across multiple users. personal and ubiquitous computing. Personal and Ubiquitous Computing **7**, 275–286 (10 2003). https://doi.org/10.1007/s00779-003-0240-0 [Cited on page 6.]

[3] Asterisk: `https://www.asterisk.org/`, accessed: 2024 [Cited on page 22.]

[4] Breiman, L.: Random forests. Machine Learning **45**, 5–32 (10 2001). https://-doi.org/10.1023/A:1010950718922 [Cited on pages 6 and 15.]

[5] Bunse, M., Jakobs, M.: Critical difference diagrams with python and tikz. `https://mirkobunse.github.io/critdd/index.html`, accessed: 2024 [Cited on page 19.]

[6] Demsar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research **7**, 1–30 (01 2006) [Cited on page 19.]

[7] Eberstein, V., Sjöblom, J., Murgovski, N., Haghir Chehreghani, M.: A unified framework for online trip destination prediction. Machine Learning **111**, 1–27 (07 2022). https://doi.org/10.1007/s10994-022-06175-y [Cited on page 6.]

[8] Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. p. 226–231. KDD'96, AAAI Press (1996) [Cited on pages 6 and 14.]

[9] Filippou, S., Tsiartas, A., Hadjineophytou, P., Christofides, S., Malialis, K., Panayiotou, C.: Improving customer experience in call centers with intelligent customer-agent pairing (05 2023) [Cited on page 8.]

[10] Google Cloud: Cloud computing services. https://cloud.google.com/, accessed: 2024 [Cited on pages 2 and 24.]

[11] Google Cloud: Vertex ai. https://cloud.google.com/vertex-ai/, accessed: 2024 [Cited on page 24.]

[12] Guo, G., Wang, H., Bell, D., Bi, Y., Greer, K.: Knn model-based approach in classification. In: Meersman, R., Tari, Z., Schmidt, D.C. (eds.) On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE. pp. 986–996. Springer Berlin Heidelberg (2003) [Cited on pages 6 and 15.]

[13] Hauska, H., Swain, P.: The decision tree classifier - design and potential (02 1975) [Cited on pages 6 and 15.]

[14] IBM: Crisp-dm help overview. https://www.ibm.com/docs/en/spss-modeler/18.5.0?topic=dm-crisp-help-overview, accessed: 2024 [Cited on pages vii and 10.]

[15] Kafka: https://kafka-python.readthedocs.io/en/master/index.html/, accessed: 2024 [Cited on page 22.]

[16] Koole, G., Mandelbaum, A.: Queueing models of call centers: An introduction. Annals of Operations Research **113**, 41–59 (07 2002). https://doi.org/10.1023/A:1020949626017 [Cited on page 2.]

[17] Liao, C., Chen, C., Xiang, C., Huang, H., Xie, H., Guo, S.: Taxi-passenger's destination prediction via gps embedding and attention-based bilstm model. IEEE Transactions on Intelligent Transportation Systems **PP**, 1–14 (01 2021). https://doi.org/10.1109/TITS.2020.3044943 [Cited on pages 7 and 12.]

[18] Lloyd, S.: Least squares quantization in pcm. IEEE Transactions on Information Theory **28**(2), 129–137 (1982). https://doi.org/10.1109/TIT.1982.1056489 [Cited on pages 6, 7, and 14.]

[19] Mamajek, E.E., Torres, G., Prsa, A., Harmanec, P., Asplund, M., Bennett, P.D., Capitaine, N., Christensen-Dalsgaard, J., Depagne, E., Folkner, W.M., Haberreiter, M., Hekker, S., Hilton, J.L., Kostov, V., Kurtz, D.W., Laskar, J., Mason, B.D., Milone, E.F., Montgomery, M.M., Richards, M.T., Schou, J., Stewart, S.G.: Iau 2015 resolution b2 on recommended zero points for the absolute and apparent bolometric magnitude scales. arXiv 1510.06262 (2015) [Cited on page 18.]

[20] Mehrbod, N., Grilo, A., Zutshi, A.: Caller-agent pairing in call centers using machine learning techniques with imbalanced data. In: 2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC). pp. 1–6 (2018). https://-doi.org/10.1109/ICE.2018.8436314 [Cited on page 7.]

[21] Moreira-Matias, L., Gama, J., Ferreira, M., Mendes-Moreira, J., Damas, L.: Predicting taxi–passenger demand using streaming data. IEEE Transactions on Intelligent Transportation Systems **14**(3), 1393–1402 (2013). https://-doi.org/10.1109/TITS.2013.2262376 [Cited on page 6.]

[22] Moreira-Matias, L., Nunes, R., Ferreira, M., Mendes-Moreira, J., Gama, J.: On predicting a call center's workload: A discretization-based approach. In: Andreasen, T., Christiansen, H., Cubero, J.C., Raś, Z.W. (eds.) Foundations of Intelligent Systems. pp. 548–553. Springer International Publishing, Cham (2014) [Cited on page 7.]

[23] van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A.W., Kavukcuoglu, K.: Wavenet: A generative model for raw audio. CoRR **abs/1609.03499** (2016), http://arxiv.org/abs/1609.03499 [Cited on page 2.]

[24] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., Louppe, G.: Scikit-learn: Machine learning in python. Journal of Machine Learning Research **12** (01 2012) [Cited on pages 14 and 17.]

[25] Robusto, C.C.: The cosine-haversine formula. The American Mathematical Monthly **64**(1), 38–40 (1957), http://www.jstor.org/stable/2309088 [Cited on page 14.]

[26] Rousseeuw, P.: Rousseeuw, p.j.: Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. comput. appl. math. 20, 53-65. Journal of Computational and Applied Mathematics **20**, 53–65 (11 1987). https://doi.org/10.1016/0377-0427(87)90125-7 [Cited on page 18.]

[27] Shearer, C.: The crisp-dm model: the new blueprint for data mining. Journal of data warehousing **5**(4), 13–22 (2000) [Cited on page 9.]

[28] Zong, F., Tian, Y., He, Y., Tang, J., Lv, J.: Trip destination prediction based on multi-day gps data. Physica A: Statistical Mechanics and its Applications **515**, 258–269 (2019). https://doi.org/10.1016/j.physa.2018.09.090 [Cited on page 7.]