FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Automatic definition and classification of points of interest of vehicles through floating car data

Tiago Alexandre de Sousa Dias da Silva

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: João Pedro Mendes Moreira, PhD

Company Supervisor: Leonel Rocha Araújo, MSc

July 31, 2020

Automatic definition and classification of points of interest of vehicles through floating car data

Tiago Alexandre de Sousa Dias da Silva

Mestrado Integrado em Engenharia Informática e Computação

July 31, 2020

Abstract

Nowadays there is a growing interest in the collection and extraction of relevant information from data. With this in mind, variables describing the operation of distribution vehicles can be retrieved in real time with the use of tracking systems. From the analysis of the data gathered, there is then an interest in identifying the points of interest in the trip of these delivery vehicles, such as the loading and unloading of cargo, maintenance or refueling. With the identification of these operations, fleets managers can get exact information about the vehicle state in certain points, along with the creation of more detailed reports and easier resource management. This automatic labelling process can also help in adding a layer of intelligence to automotive systems, as the final objective of a vehicle's trip can be identified beforehand.

The approach to solve our problem can then be divided into two tasks: an identification task and labelling task. Various clustering methods are ran against processed raw circulation data to first find the points of interest, with the output being evaluated considering the context, determining where the vehicle stops more and for longer periods of time. To then give meaning to this first task, multiple classifications models are created to label the points found, with each model's accuracy being measured to determine which algorithm has the capability to better predict the reason for a vehicle stopping.

The clustering method used to create the points of interest for the predicting task was found to highly adapt to the spatial data, which is corroborated by the comparison of the mapping of its output with a known point of interest dataset. Many classification algorithms also achieved high accuracy measurements, being attained a value as high as 97% in the case of one of the companies analysed.

With all this in mind, it can be said that the approach adopted produced a positive outcome, establishing a whole process from raw circulation data to automatically created and labelled points of interest in vehicle distribution fleets. By implementing this solution in live scenarios, transportation and logistics companies can not only analyse the purpose of all stops of the vehicles in their fleet, enabling routing optimization, but also get a constant and reliable log of their geographical carbon footprint and its compatibility with the ever-increasing low emission zones.

Resumo

Atualmente existe um interesse crescente na recolha e extração de informação relevante a partir de dados. Com isto em mente, variáveis descrevendo o funcionamento de veículos de distribuição podem ser recolhidas em tempo real com a utilização de sistemas de rastreio. A partir da análise dos dados recolhidos, há então interesse em identificar os pontos de interesse na viagem destes veículos de entrega, tais como a cargas e descargas, manutenção ou reabastecimento. Com a identificação destas operações, os gestores de frota podem obter informações exatas sobre o estado do veículo em pontos específicos, juntamente com a criação de relatórios mais detalhados e uma gestão mais fácil dos recursos. Este processo de etiquetagem automática pode também ajudar a acrescentar uma camada de inteligência a sistemas automóveis, uma vez que o objetivo final da viagem de um veículo pode ser determinado previamente.

A abordagem para resolver o nosso problema pode ser então dividida em duas tarefas: uma tarefa de identificação e uma tarefa de rotulação. Vários métodos de agrupamento são executados usando dados brutos processados para primeiro encontrar os pontos de interesse, sendo o resultado avaliado considerando o contexto, determinando onde o veículo para mais e por períodos de tempo mais longos. Para então dar sentido a esta primeira tarefa, múltiplos modelos de classificação são criados para etiquetar os pontos encontrados, sendo a precisão de cada modelo medida para determinar qual algoritmo tem a capacidade para melhor prever a razão de paragem de um veículo.

O método de agrupamento usado para criar os pontos de interesse para tarefa de previsão mostrou alta adaptabilidade aos dados espaciais, o que é corroborado pela comparação do seu resultado com uma lista de pontos de interesse conhecidos. Vários algoritmos de classificação também alcançaram valores elevados de precisão, sendo atingido um valor tão alto como 97% no caso de uma das empresas analisadas.

Com tudo isto em mente, pode ser dito que a abordagem adotada produziu um resultado positivo, estabelecendo um processo completo desde dados brutos de circulação até pontos de interesse automaticamente criados e etiquetados em frotas de veículos de distribuição. Ao implementar esta solução em cenários ao vivo, empresas de transporte e logística podem não só analisar o propósito de todas as paragens dos veículos da sua frota, permitindo otimização de rotas, mas também obter um registo geográfico constante e fiável da sua pegada de carbono e a sua compatibilidade com as zonas de baixas emissões que são cada vez mais comuns.

Acknowledgements

First of all, I would like to express my gratitude to AddVolt and all its collaborators for their hospitality during the thesis period along with the data that made this work possible.

Secondly, a special thanks to my supervisors, Professor João Pedro Mendes Moreira and Leonel Rocha Araújo, who guided every step of this work in the right direction and were always available to answer every doubt along the way.

Lastly, to my family and friends as well as everyone that helped me reach this stage.

Tiago Silva

Contents

List of Figures

List of Tables

Abbreviations

FCD	Floating car data
GPS	Global Positioning System
POI	Point of Interest
KDD	Knowledge discovery
DBSCAN	Density-based spatial clustering of applications with noise
OPTICS	Ordering points to identify the clustering structure
AUC	Area Under Curve
SNN	Shared Nearest Neighbour
API	Application Programming Interface
UTM	Universal Transverse Mercator

Chapter 1

Introduction

The aim of this chapter is to give a contextualization for the problem along with the interest in addressing it. The objectives of the research and the content of this document are thoroughly defined.

1.1 Context

Urban distribution vehicles base their operations around their delivery capabilities. Tracking systems are then used to support these deliveries by gathering data pertaining to their trips. With the advance in computing technology, these systems have become more capable and affordable, making it possible to be implemented in a larger scale. These tracking systems bring about many advantages for distribution vehicles as they make it possible for the optimization of routes taken, better resource management and more accurate delivery times.

The gathered data is collected every second while the vehicle is in movement and this data contains many details about the vehicle's operation such as position, velocity or fuel available. It is saved in a time series format, more specifically called floating car data (FCD), as the data is being collected by a sensor contained in the moving vehicle itself as opposed to being positioned in a stationary location.

As the evolution seen in tracking systems made them more precise with the advance in years, the same can be said regarding capabilities to store, analyse and process data. Increased computing power technology lead to a higher feasibility and capability in saving large quantities of data as storage units became cheaper in price while also increasing storing capacity. These advances in computing power also contributed to being able to run more complex algorithms and at the same time handle larger quantities of data with less concerns about computational cost, opening up new possibilities in the extraction of relevant knowledge from data.

One of the companies to identify an opportunity in this scenario is AddVolt which has as their main value proposition the electrification of refrigerated chambers used in the transport of temperature sensitive goods. AddVolt maintains a connection of its roadside systems to its servers, transmitting data via GSM (Global System for Mobile communication) similar to a IoT (Internet of

Things) system. Through this connection, several variables related to the operation of the vehicle, cold chamber and AddVolt's system are collected for analysis and reporting.

With the possibilities opened up from these standpoints, an opportunity emerges for the understanding of a distribution vehicle's operation through the analysis of the data gathered from the tracking systems.

1.2 Motivation

During the process of a delivery from a distribution vehicle, many different situations can occur as the vehicle operates. There is an interest in identifying those situations such as loading and unloading cargo, maintenance or refuelling. To achieve that, there is a need in analysing the data gathered during the vehicles circulation and, from there, to try and find meaningful patterns in the circulation that make sense in the context of a distribution vehicle.

Due to the frequency of the capture of the vehicle's position, a large amount of data is generated which already in itself presents a great potential for the understanding of the vehicle's delivery patterns[28].

From this data analysis, there is an opportunity for automatically labelling vehicle stopping moments, which is a process done manually otherwise. While there is many efforts in creating projects that label points, both free such as OpenStreetMaps[1] or proprietary such as Google Maps[5], this is not enough to satisfy the categorization of the points for a distribution vehicle, as a same point can represent a point of cargo loading for one company and unloading for another. So, it is important to present a customized solution for each company which completely depends on the data generated during its circulation.

From the identification of these operations from FCD come many benefits. The use of fleet management and vehicle tracking systems has been increasingly widespread, being present in six million vehicles in North America and approximately five million in the European Union in 2009[13], meaning a solution created from raw FCD has high value since it can be applied to a high number of systems. Knowing exactly where the vehicle is located at and getting a description beyond the position of it will help in the management of resources and provide more detailed freight reports. Beyond that, the vehicle's compatibility with low emission zones can be determined, as this solution can also encompass the gauging of the carbon footprint in a given geographic location. Low emission zones are areas where more polluting vehicles either cannot enter or have to pay for that entrance, as a measure to improve air pollution[6], which increases the interest in this analysis given the higher value imposed nowadays in ecological practices by governing bodies.

1.3 Goals

The main aim of this project is essentially a knowledge retrieval task from FCD, being the final objective representing the labelled points of interest (POIs) for a distribution vehicle in a way that

is simple to interpret, being the most simple way to do that by representing these identified POIs in a map. Two main tasks can be identified from these objectives: the identification of these points of importance in the vehicle's operation followed by its labelling.

More specifically, before beginning the first task, the knowledge retrieval process encompasses the analysis of the data available. There is a need to understand how the vehicles move on the road in order to identify where we should be looking for candidate POIs by considering how the vehicle behaves in a certain area. For example, if a vehicle consistently moves at a high speed over a certain area, that area can be discarded as POI, while if a vehicle stays for a long period of time in a certain area and it happens frequently over many days it should be paid special attention.

The first task of the project is grouping the points of the vehicle's circulation to determine the actual POIs, yet to be labelled. This grouping should be tested using different setups and approaches in order to find what works best with FCD and be evaluated in strict format as it sets the base for the next task in the project.

Once the POIs are determined, an additional feature engineering task needs to be carried out to gather information that can help label these POIs. This additional task is needed as gathering POI and labelling them have very distinct requirements data-wise. There is a need to give meaning to the first task by labelling the points found in order to complete the whole process.

1.4 Document Structure

This document is divided in chapters and structured as described in the following statements.

In chapter 2, the state of the art is presented. It starts by introducing data mining concepts and particular applications to time series. Finally, different solutions to similar problems as one the being considered are presented.

In chapter 3, the problem and a solution for it is presented, along with the tasks needed to reach that solution.

In chapter 4, the approach taken for the identification task is explained in detail, building the basis for the next task, by finding the POI in the vehicle's circulation.

In chapter 5, the results from the previous chapter are used to build models using classification algorithms, which assign labels to POIs, therefore completing the process.

To close it out, chapter 6 presents conclusions pertaining to the document as a whole, its knowledge contribution and possible future work.

Chapter 2

A review on Machine Learning applied to Time Series

In this chapter, a review on the concepts relevant to this dissertation is presented. It starts out by presenting data mining and machine learning. Following that, a more particular analysis on the time series and its applications is performed, along with possible data mining tasks that can be performed over this data format. Finally, related work to the problem that will be addressed is presented along with the approaches taken.

2.1 Data Mining

Data mining is frequently defined as the search for interesting patterns in data. Data mining and knowledge discovery (KDD) are very intertwined concepts. From the huge amount of data available nowadays it comes a need for tools capable of automatically uncovering relevant information from that data and transform it into organized knowledge. It was from this need that the concept of data mining was born, being the central concept in KDD[32].

To guarantee that the data mining process does not find meaningless or invalid patterns, other additional steps in KDD process are essential to ensure that appropriate knowledge is retrieved. The proper preparation and selection of the data along with correct interpretation of the results of the mining present an equally important task[23].

2.2 Machine Learning

Machine learning is a fast-growing area of study and, like pattern recognition and statistics, it is one the technologies that supports data mining in the process of finding relevant knowledge. Its main objective is for computers to learn how to automatically extract complex patterns from data and use those patterns to make decisions[32]. Machine learning tasks are essentially divided into two major groups, supervised and unsupervised, which concepts are discussed more in depth in section 2.4.

2.3 Time Series Data Mining

Data can be captured in multiple ways which also leads to different representations and organizations in the way the data is stored. In the case of time series, measurements performed sequentially over a period of time capture the value of variables at a certain day and hour. Due to the way they are stored, it gives them an importance in being analysed as a whole instead of simply analysing individual entries[25]. Therefore, small fluctuations in data should be ignored and the retrieval of relevant knowledge should focus in finding a pattern in a established time scale so that sequences of time measurements can be compared. Its main characteristics are then its size, high dimensionality and the fact that they update continuously.

From the nature of the time series, many challenges arise[21]. High dimensionality along with growing digital sources of information creates an issue with data representation as there is a need to reduce the dimensionality of the data while keeping fundamental shape characteristics. Other problem arises from the comparison of time series sequences. As previously mentioned, time series should be analysed as a sequence instead of an individual entry, and from there comes the question of how should that analysis match or distinguish between two time series sequences.

The analysis of time series can have many objectives[31]. Forecasting, in which by analysing past sequences in time series, it may be possible to predict the values for future sequences. It is possible to use it in the production control processes in order to detect any anomaly. They can also be used to determine different types of patterns in data[35]. By monitoring the fluctuation of the prices of goods monthly, seasonality patterns can be extracted, such as in cases where certain products can only be produced under a certain weather. Trends can be observed by noticing a lasting increase or decrease in the data over time, like in the case where a constant increase can be noticed in global temperature. Finally, cycles can be detected when rise and falls are detected without a explicit time frame, as seen in stock markets. These features in the data are essential to identify so the right predictions can be made.

2.4 Machine Learning Tasks

To best extract relevant knowledge from data, different approaches must be tested, as different problems call for different solutions. In this section, different machine learning tasks are introduced and then their application to time series is acknowledged by referencing implementations using those types of tasks.

2.4.1 Unsupervised Learning

While it encompasses other learning methods such as association rules, unsupervised learning is typically used as a synonym for clustering. The learning process is then unsupervised as classes are found directly in the data from input examples[32], being this class label information typically not present. For this reason, it is considered as a form of learning by observation as opposed to

learning by examples, which means many times it can be hard to determine the success of such learning method as there is not a direct measure of success [33].

2.4.1.1 Clustering

The objective of clustering is the division of data into subsets. These subsets are referred as clusters, where each object inside a cluster is similar between each other while different from the objects in another cluster [32]. An example of the result of a clustering task is shown in figure 2.1.

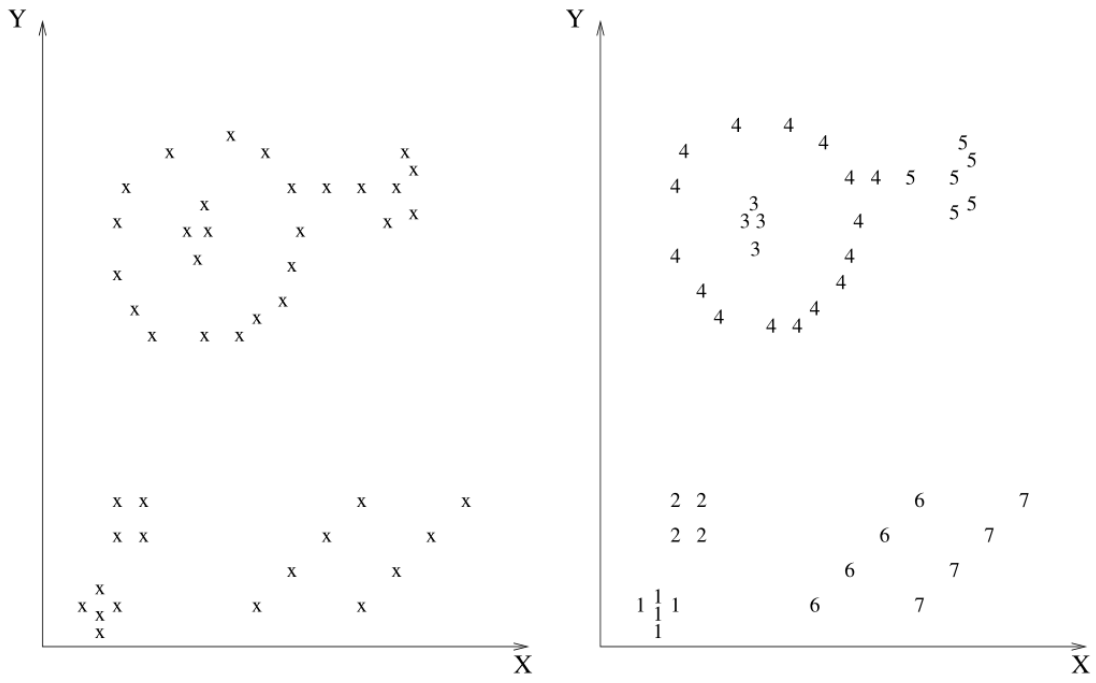

Figure 2.1: An example of a data clustering task[36]. In the left, data is represented before clustering is performed. After the clustering algorithm has ran, each data point has a number assigned, being this number the cluster this data point belongs to.

A clustering task normally starts out by the identification of the best features available, which means choosing the subset of data fields that will lead to obtaining the most accurate results, being this subset for each input example considered a pattern. An accurate result in this context translates normally into data partitions which the user can interpret to some level. New features might also be generated by transformation of these original data fields constituting the input. Next, it is important to define the best measure to compare the proximity between patterns, normally gauged by a distance function. This distance function is then used to group patterns into clusters[36], depending on other constraints based on the algorithm chosen. Optimizations can be made by fine tuning every step, starting from the feature selection. A result is always generated even if no actual grouping is present, giving the cluster validation step a very important role[50].

According to [32], clustering methods can be briefly classified as:

- Partitioning methods: a n set of objects is divided into a k number of clusters where $k \leq n$.
- Hierarchical methods: can be divided into agglomerative or divisive methods, depending in how the hierarchical decomposition is built. The agglomerative method starts from a single object representing a cluster and, from there, other objects or clusters are joint depending on how close they are, or a termination condition is reached. In the divisive method, a single cluster containing all objects is divided into smaller clusters until all clusters are left with one object or a termination condition is reached.
- Density-based methods: can adopt any kind of shape. As long as there are enough objects in the vicinity the cluster will keep growing, being the number of objects and the radius of this vicinity normally set by the user.
- Grid-based methods: the object space is divided into a grid where objects in a same cell will belong to the same cluster.

2.4.1.2 Clustering in Time Series

The time series clustering task can be divided into two sub-tasks[21]. Clustering can be performed over each complete time series in a set, being the objective to regroup complete time series into clusters so that they are as similar to each other as possible inside each cluster. This is called whole series clustering. In the other sub-task, subsequence clustering, clusters are built by using a part from a single or multiple longer time series.

The research of clustering in time series has been under development over the years and in many domains. For instance, in [16], in order to detect injuries from sets of time series, pattern recognition is performed by matching equal patterns with Euclidean distance. Another application can be found in [26], where self-organizing maps are used for pattern discovery in stock data. To find abnormal time series subsequences, [37] uses Euclidean distances to measure similarity between examples, having this work many applications in real life contexts such as medicine, in the subject of electrocardiograms. Clustering algorithms like k-means are used in [51] to find intrusions in networks with a high detection rate. Finally, [39] surveys clustering techniques to analyse gene expression data. From these works presented, there is a clear conclusion that the areas of application of clustering techniques to obtain knowledge from time series is immense, highlighting their worth in being pursued.

Having formed a general idea about clustering and its role established in time series, we can then choose important requirements for a clustering algorithm in order to help determining what algorithm will perform better in the context of our problem where the data is in a time series format. Using the criteria established in [32], the algorithm chosen should be able to:

- Discover clusters with an arbitrary shape.
- Deal with noisy data.
- Incorporate new data without having to be reran completely from scratch.

- Cluster high-dimensionality data due to the nature of time series.
- Produce a result that can be easily interpreted and usable in order to achieve later tasks.

2.4.2 Supervised Learning

The main goal of supervised methods is to learn the relation between input variables, independent attributes, and a target attribute, known as a dependent variable. It is considered supervised as the learning of that relation comes from labeled examples which are already partitioned into classes. From these examples used as training, these relations can be mapped into models. These models allow us then to conclude on a class given certain input variables for new examples. The two main supervised models are classification models and regression models[40].

2.4.2.1 Classification

In the case of a classification task, a user-defined notion of grouping is obtained from the training set, contrary to the case seen in clustering. Typically, a classification task goes through two phases denominated training phase and testing phase. In the training phase, a model is generated from the training examples, being this model capable of offering a prediction for a new entry with no class, which is what occurs in testing phase, where unseen test instances are fed into the model to be assigned a predicted class[14]. Having the predicted classes assigned to the test examples, it is essential to determine how accurate the model is in grouping these test examples. To test how accurately these test examples will be predicted, one possible approach is splitting the training examples into a validation set which is used to determine prediction error for the model built[33]. The examples used in the validation set should not be included in the training set in order to obtain an accurate value of the rate of prediction of the model. Figure 2.2 shows the progress of a classification task.

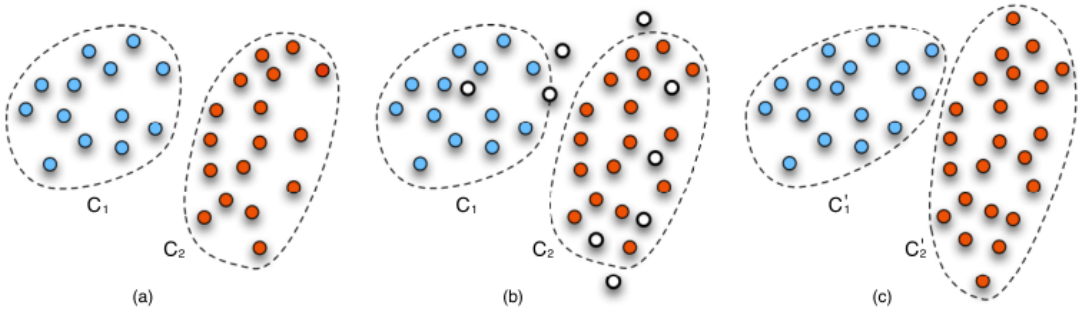

Figure 2.2: An example of a data classification task. In (a), a dataset with two classes (C1 and C2) is presented, which will be learnt by an algorithm. In (b), new examples with unassigned classes are introduced which will be predicted by the model resulting in the class distinction presented in (c)[21].

One of the most vitals points in a classification task is determining the capacity of a model to generalize, in order to be able to predict unseen instances correctly[32]. Training data that is too big or small might lead the model to learn incorrect deductions if not trained correctly. It might adapt too much to the training examples leading it to a situation where it learnt only to predict properly for the training instances, called overfitting. Likewise, there might not be enough examples for the model to learn the underlying structure in the data, if there is any, which leads it to a situation of underfitting, where the model simply did not get sufficient training samples to learn how to predict correctly unseen test instances.

2.4.2.2 Classification in Time Series

Classification in time series works similarly as explained in 2.4.2.1. A training set of time series with a labelled class is used for training the classifier, which is then used to predict the class for new time series[21]. Even if the approach is similar to one in a traditional classification task, special attention should be paid due to the nature of the data[25].

There are many examples of works in the study of classification for time series. [38] proposes a representation for time series with focus in efficient computation of the data that can be used for both classification and clustering. In [27], a technique is described for finding patterns relevant to solving classification problems in time series. Due to the lack of a large amount of labeled training data in specialized domains, [48] proposes a semi-supervised technique to build accurate time series classifiers while using a reduced amount of labeled examples. Finally, [49] offers a approach to reduce the numbers of examples used for training while maintaining accuracy in the case of a one-nearest-neighbor classifier with Dynamic Time Warping in order to improve computational costs.

2.5 Floating car data

While the analysis so far has focused on time series, it is also worthy to present FCD as a type of time series data in order to provide a better context to the problem that will be addressed. FCD is based on the transmissions from GPS sensors in the vehicle while in movement, providing access to fields that characterize the movement of the vehicle such as latitude, longitude, altitude or speed. Each of the data entries containing these fields comes timestamped with the moment these measurements were performed, being sent at a fixed interval in real time to a system that will store the data.

This type of data represents a reliable and cost-effective way to collect accurate traffic data, representing a good addition to fixed-point traffic sensors which involve high installation and maintenance costs[20]. A few examples of the research using FCD are the deduction of traffic speeds such as in [20], estimation of a route's travel time[44] and incident detection[34].

2.6 Automatic definition and classification of POIs

The objective of this dissertation project is finding POIs related to a vehicle's operation and the categorization of that operation through the analysis of FCD. To better define a proper approach to solve the problem, similar works must be analysed. Our main point of focus are the approaches that base their work around the use of data mining methods to both find and categorize POIs.

In [46], the authors aim for the identification of location contexts using POI data. Location contexts constitute a well defined geographic region with a description, characterizing the area the person is in. Using a POI database, clustering is performed over this data by taking advantage of the coordinate fields which form a "cloud" in the geographic space. The Shared Nearest Neighbour algorithm is used, which constitutes a spatial density-based algorithm and a good option in this context as it can identify clusters of any shape, density or size.

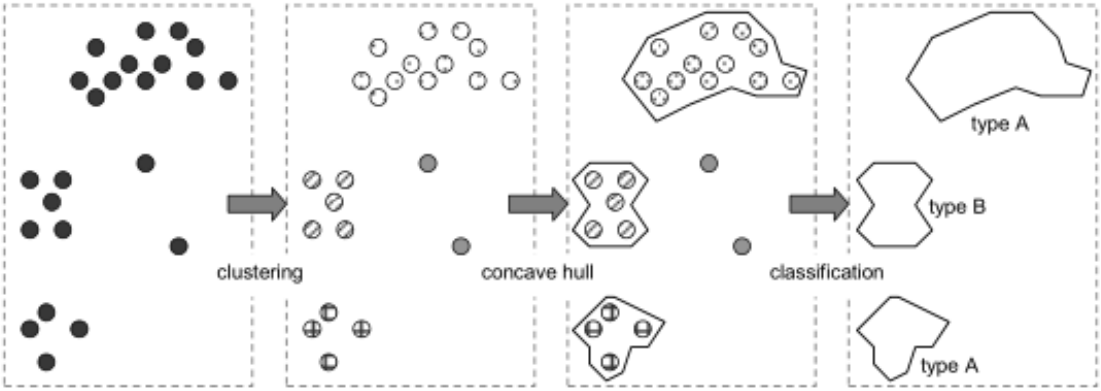

Figure 2.3: Clustering process performed in [46] to identify location contexts. Having a group of POIs, a clustering algorithm is ran to group them. The position of each POI in the cluster determines the area that the location context will occupy. The last step is then to categorize these locations into different types such as A or B, in the context of this figure.

The testing of the output for the clustering algorithm was done by overlapping the results with satellite images from Google Earth[7] which is also used for the classification task, as clusters were manually labelled in order to produce data to train and test decision trees. As the dataset created was unbalanced, part of the examples in classes had to be duplicated, obtaining a final result with 92% accuracy.

[30] has the goal of labelling the purpose of a trip using GPS data. In this case, the POIs are trip stops (with a minimum duration of five minutes) and were obtained by the use of a density-based algorithm, DBSCAN. This kind of algorithm was chosen as it can detect clusters with arbitrary shapes and works well with data having spatial features such as coordinates. An important aspect is that, from the entries that compose a cluster, aggregations like averages were performed over the data, using the attributes "time of arrival" and "length of stay" to create new features for a POI, which then can be used for the classification task. Decision trees were used to identify the trip purpose after training the model using the POIs that were found and manually labelled.

Using GPS data along with a travel diary partly filled by the users themselves, [41] has the objective to further automatize travel imputation in order to improve travel purpose identification in general and make the creation of these surveys a easier and more accurate task, facilitating the burden on the users. To achieve this, the authors begin by identifying the different types of travel purpose and then locations are clustered by activities using an agglomerative hierarchical clustering approach. Random forests are then used to identify trip purposes such as "being home", "working" or "recreational activities", obtaining an overall mean accuracy of 80%. An important point to note is that the model created identified common activities more accurately, being speculated by the authors that this due random forests favoring classes that appear more in the case that a good model can not be built.

From continuous GPS trajectories collected using mobile phones, [29] aims to firstly identify moving and stopping points in these trajectories, and then, in the stopping points, distinguish activity points from non activity points. The data collected has as important features geolocation, a time stamp and signal quality so records gathered with low signal can be discarded. Trip starts, ends and trip purposes are inputted by the mobile phone owners. The first step was identifying moving and stopping points and to this end C-DBSCAN was utilized, a variation of DBSCAN. While DBSCAN is capable of identifying stop points as clusters using the geographical data, it did not adapt properly to this context, as this algorithm was developed with no temporal notion and, due its definition, also created incorrect clusters under movements in a straight road at low speed. To solve these problems, C-DBSCAN was created by adding a time sequence constraint and a direction change constraint to DBSCAN. Having the clusters representing stopping points, identifying them as activity points or not was the next step. The algorithm chosen for this task was support vector machine (SVM), having as main features stop duration, the mean distance from the points in a stop cluster to its centroid and the minimum distance from that stop position to home and work. Stop points were obtained with a 90% accuracy and activity points were distinguished with a 96% accuracy.

Once again from the collection of GPS data and a travel diary, [52] has the objective to identify trip ends as they are crucial points to separate trajectories into journey and activity segments. Firstly, a density based spatio-temporal algorithm is applied to find candidates for stops. This algorithm is proposed by the authors and has the objective to answer the missing temporal notion in algorithms like DBSCAN, just like [29], but in this case by using a sliding temporal window. Trip ends are then identified by the use of a random forest classifier by distinguishing actual trip ends from stops with a very satisfactory accuracy of 99.2%.

Finally, [47] focuses on stop purpose classification from GPS on the context of commercial fleets, having then a similar context to the one this dissertation project is inserted in. More specifically, it intends in specifying the purpose of a stop as work related (delivery) or non work related (maintenance or refuelling, as examples) giving fleet managers access to more complete information about the vehicles' trips. For vehicle stop detection, the authors developed their own spatio-temporal clustering algorithm that, firstly, associates GPS data entries to a type that characterizes the vehicle's movement and then gathers them into groups using these types. Stops are defined

by chronologically consecutive types "engine off" or "idling" while movements are determined as type "journey". With this method they were able to properly identify places where the vehicles were stopped and also where operations were done. For the creation of features for the classification task, an extensive list of new features was computed, divided into four groups: stop-wise features which were obtained by the aggregation of the GPS entries belonging to each stop cluster; POI features such as "distance to closest fuel station" or "distance to closest repair facility" were obtained by checking a radius around the obtained stop location using an external source containing POI data; stop cluster features which had fields that characterized the area around a given stop; finally, sequential features which characterized the consecutive stops of a user in a day. These features were then used to create a random forest classifier to identify stops as work related or non work related, obtaining a AUC score of 0.931. The authors also ranked the most relevant features for the prediction in this classifier, the top five being: average stop duration in cluster, stop duration, total time with engine off, maximum time with engine off and max stop duration in cluster. Given the context of this work, the analysis of all the features used and its relevancy is particularly interesting for solving our own problem.

From the analysis of these works, options for both the definition and classification of POIs were found. DBSCAN was the most favoured algorithm to form clusters that adapted to the different contexts, but also presented disadvantages such as not having a temporal notion leading to points being identified incorrectly. For the labelling of the points found, Random Forests constituted a solid answer, but it could be noticed that the most important step in the classification tasks was the creation of new features from the data composing the clusters along with the use of external sources to enrich the existing data.

Chapter 3

Solution Proposal

The goal of this dissertation project is automatizing the process of identifying and labelling the POIs of an urban distribution vehicle. From this project summary, two tasks can be clearly delimited: identification and labeling. From the identification task, various clustering algorithms will be tested over the dataset from which non labelled POIs will be obtained. To give meaning to this first task, there is an interest in labeling the points found with the operation performed by the vehicle with the creation of a classifier which is capable of assigning a meaning to unlabelled POIs. The following sections will go into more detail about each task and then the datasets are described along with the features they contain.

3.1 Identifying the POIs

The first step in this project is the identification of the points of interest in the circulation of various vehicles belonging to a same company. The intent of this step is then determining automatically where the vehicles normally stop. A POI is then represented by a pair of coordinates where the vehicle either stops very frequently or stays for long periods of time, meaning it holds some kind of importance to the vehicle's circulation routine.

Due to the frequency of the data transmission process, a large amount of data is gathered, creating a need to initially perform a downsampling operation so longer periods of data can be analysed with a lower computational burden. This can be achieved by aggregating data with the use of summary statistics. This process is not enough to prepare the data, being there also a subsequent step to filter incorrectly gathered data, making it possible for the following processes to be more precise.

Having the data processed, POIs are then identified by the use of clustering algorithms. The result of these algorithms are points represented by geographical coordinates (latitude, longitude), making it possible to represent the output of each algorithm in a map, which helps the analysis being a more objective process. Nevertheless, these results also have its outcome analysed by the use of business knowledge, as the points obtained are compared to existing known POIs so a better grasp of these algorithms performance can be determined.

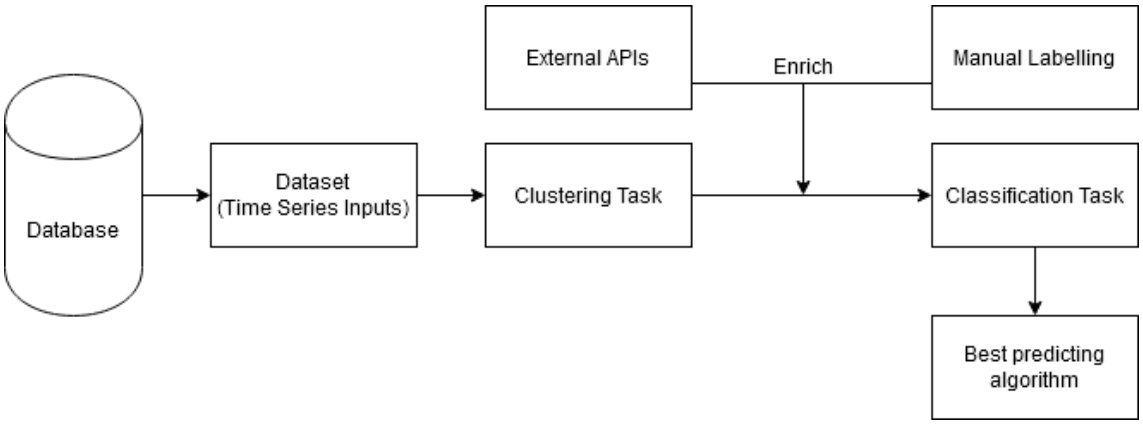

Figure 3.1: Proposed solution and the two main tasks: clustering and classification.

3.2 Labelling the POIs

Having the POIs identified, the following step is to determine how they should be labelled. By using the data generated by the clustering algorithm as input, the POIs are given a identifying label, determining the purpose for stopping of that vehicle in the context of the company's operations. To this end, the training of a model capable of predicting these labels is needed.

The first step is then generating features that can be fed to the various classification algorithms. From the result of the clustering algorithm, new features were already computed, as a cluster is built from the aggregation of the entries of a vehicle's circulation. In addition to that, by the use of OpenStreetMaps[1]'s API, features identifying amenities, shops, buildings and other infrastructure surrounding the clusters were also obtained, giving more alternatives to the algorithms to identify patterns in the data.

Finally, by the use of the result of feature engineering, the data is split into train and then test so as to create and determine the accuracy of the classifiers trained, being this data previously manually labelled through an interface available to the fleet manager and completed with business knowledge. Two different approaches to dataset creation are tested and compared, one where each entry represents a POI and the other where each entry represents a stop of the vehicle in that POI, with the final objective of determining which produces more accurate models.

The result from this process is then a classifier capable of predicting the purpose of a vehicle's stop in a certain point, such as unloading cargo at a client or refueling the vehicle, being the classifier chosen the one with best accuracy in predicting these labels.

3.3 Data sources and base datasets

Vehicle's circulation data is gathered by a vehicle bus module, Controller Area Network(CAN bus), and is sent every second to a remote server which stores the data in a PostgreSQL[8] database. This data reflects the movement of the vehicle over time, gathering various different metrics on

both the vehicle state and its refrigeration system. Data from the circulation of the vehicles from two different companies will be analysed to better test the solution presented as different companies will have different clients, warehouses and operate in distinct areas, meaning that there is a higher security in the solution described in the case that both reach satisfactory results.

The data for each company is organized by what the module gathers in a given week, there being circulation data from October 2017 to February 2020 belonging to different vehicles. A file for each week normally ranges from 100000 to 600000 lines, occupies from 15 megabytes(MBs) to 115 MBs and contains 32 columns, being the most relevant to solve the problem at hand presented in table 3.1.

By performing an initial analysis, it could be identified that this base dataset contains essential entries for both the clustering and classification task. The clustering task will depend in the geographical coordinates represented by the latitude and longitude, while the classification task might find use in the other columns as they currently are or after their processing, where new relevant features will be ascertained from existing ones, helping the classification algorithms in the creation of more accurate models. It can also be noticed that a few variables are not reliable for use, such as *altitude*, which is not recorded properly, and *speed*, which has a more precise alternative in *speed_can*.

Field	**Data Type**	**Description**	**Example**
imei	TEXT	Data gathering device identifier.	358480088123456
registered_at	TIMESTAMP	Date and time of the gathering of this record.	2019-10-23 09:27:09
latitude	DOUBLE PRECISION	Latitude of the vehicle's posistion.	38.8123654
longitude	DOUBLE PRECISION	Longitude of the vehicle's position.	-8.6054321
altitude	REAL	Altitude of the vehicle's position.	0.0
speed	REAL	Vehicle's speed gathered from the GPS installed.	29.0
speed_can	REAL	Vehicle's speed gathered from the CAN bus.	25.7461
rpm	REAL	Vehicle's motor rotations per minute.	1099.0
compartment_temperature_1	REAL	Vehicle's temperature in compartment 1.	-20.0
compartment_temperature_2	REAL	Vehicle's temperature in compartment 2.	3.0
compartment_temperature_3	REAL	Vehicle's temperature in compartment 3.	16.0
timer_digital_2	REAL	Vehicle's diesel refrigeration unit timer.	10.0
timer_digital_3	REAL	Vehicle's tail lift timer.	13.0
grid_available	REAL	Availability of a grid system to feed current to a refrigeration unit.	0.0
load_controller_current	REAL	Current passing from the Add-Volt system to the refrigeration unit.	201.783
created_at	TIMESTAMP	Date and time of the insertion of this record into the database.	2019-10-23 09:27:13
updated_at	TIMESTAMP	Date and time of the last update of this record in the database.	2019-10-23 09:27:13

Table 3.1: Original dataset most relevant features.

Chapter 4

Finding points of interest in the vehicle's circulation

In this chapter, the process to find POIs is described in detail. To that end, data is firstly pre-processed as to make it more usable in both this task and the following one. Subsequently, various clustering algorithms results are then shown and analysed, which also includes an evaluation method that gives a better idea of each algorithms' performance in this specific context. A process to identify the best hyper-parameters for the chosen algorithm is presented, and, lastly, conclusions about the chapter in general are delineated.

4.1 Data pre-processing

Data pre-processing is an essential step for both the clustering and classification tasks. In this context, there is a clear need to make data more convenient to work with as raw data does not always have the most intuitive values in some of the columns or the information is not as summarized as it could be.

Data gathering modules are also prone to failure, creating a necessity to filter error cases. While many algorithms have the capability to deal with such cases, identifying and removing them can give more leeway for algorithms to achieve better results.

To address these concerns, data is processed and aggregated so that algorithms do not have to deal with sensor failures and run in less time with the same results, being these procedures further explained in sections 4.1.1 and 4.1.2.

4.1.1 Data resampling

There is an interest in downsampling the data in order to make it manageable to be analysed over longer periods of time with a lower computational cost, whilst at the same time aggregating each entry to represent a time frame that makes sense given the context.

As the raw circulation data being considered constitutes observations made each second, the frequency of the data was lowered to an observation made each minute, therefore reducing the

Field	**Description**	**Based on existing features**
vehicle_is_on	Vehicle is operating.	rpm > 300
vehicle_is_driving	Vehicle is operating and moving.	speed_can > 0, rpm > 300
vehicle_is_idle	Vehicle is operating but not moving.	speed_can == 0 , rpm > 300
vehicle_is_stopped	Vehicle is not operating.	rpm == 0
tail_lift_is_on	Vehicle's tail lift is raised.	timer_digital_3 > 0
tail_lift_startup	Vehicle's tail lift was started up.	tail_lift_is_on == True AND tail_lift_is_on == False on previous entry
refrigeration_is_on	Vehicle's refrigeration is operating.	refrigeration_diesel_is_on == True OR refrigeration_electric_is_on == True
refrigeration_diesel_is_on	Vehicle's refrigeration is using diesel to operate.	timer_digital_2 > 0
refrigeration_diesel_startup	Vehicle's refrigeration using diesel has started up.	refrigeration_diesel_is_on == True AND refrigeration_diesel_is_on == False on previous entry
refrigeration_electric_is_on	Vehicle's refrigeration is using electricity to operate.	refrigeration_electric_is_on_from_battery == True OR refrigeration_electric_is_on_from_grid == True
refrigeration_electric_is_on_from_battery	Vehicle's electric refrigeration is using the battery to operate.	load_controller_current > 0
refrigeration_electric_is_on_from_grid	Vehicle's electric refrigeration is using the power grid to operate.	grid_available == True

Table 4.1: New features generated from existing columns.

dataset size significantly. This frequency of observation is also meaningful in this situation, as most vehicles stop times in POIs are in the order of minutes.

While raw circulation data has many information that can be summarized without a significant loss of detail, to perform this downsampling correctly there is still a need to aggregate existing features in a way that represents the original data as close as to what it was in the new generated data.

Using the newly created features in table 4.1, aggregation of the records becomes an easier process as fields such as *load_controller_current* and *load_controller_state* are transposed into simpler values, reflecting in this case, for example, if the electric refrigeration system was being used(1) or not(0) at the minute that the example was recorded. As explained, the final result from this process represents the vehicle's movement every minute, having the data now in a more workable time scale.

4.1.2 Data filtering and constraining

As mentioned, the data gathering modules are prone to errors, not only due to device failures, but also due to the characteristics of the location where the vehicle is. GPS signals require a direct path between the receiver and the satellites, making it possible for enclosed spaces to be enough to completely block GPS signals[9].

By checking and plotting value variation for a few select fields, in this case, *latitude*, *longitude*, *rpm* and *speed_can*, repeated error cases can be spotted and filtered out, as these will have worse consequences when compared to sporadic errors, which can be more easily detected by the noise identifying capabilities of some of the clustering algorithms.

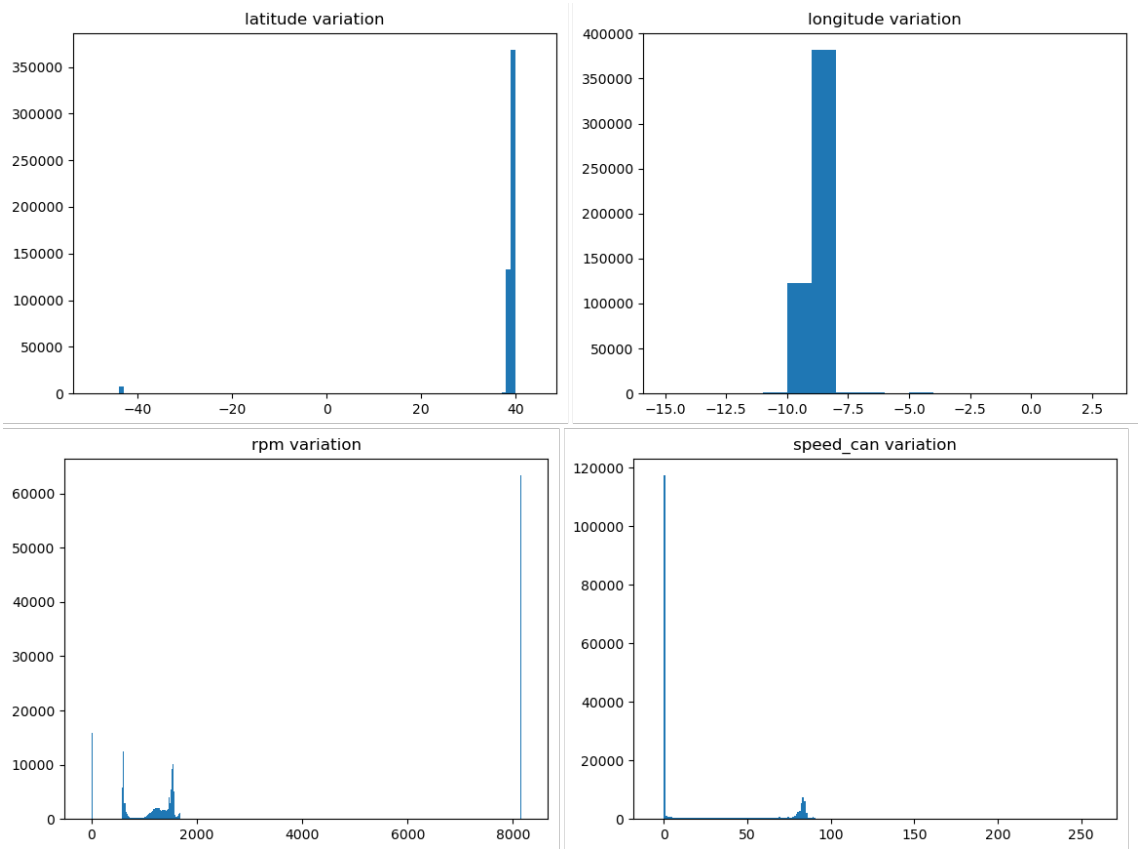

Figure 4.1: Fields *latitude*, *longitude*, *rpm* and *speed_can* value amount variation for a vehicle's circulation data corresponding to one year. Appendix A contains bigger resolution versions of the graphics presented here.

As it can be quickly noticed in figure 4.1, in the case of the *latitude* plot, a value of exactly -44 is seen a significant number of times representing a error in the data gathering of the GPS. Same can be said for the *rpm* plot where the value 8159.88 also represents an error in the vehicle's sensor due to being physically impossible. Filtering these values assures that the clustering process will obtain more accurate values, as entries with incorrect coordinates will no longer be candidates for cluster creation nor will clusters created be composed by entries with invalid RPM values.

In order to prepare it further for the clustering task, data is also constrained by the use of the features *vehicle_is_stopped* and *vehicle_is_idle* since the POIs of the vehicle will be determined by the locations where the driver stops frequently.

4.2 Clustering vehicle circulation data

With a data processing method established, we now have a base dataset which we can run the clustering algorithms against. The objective of this clustering process is finding the POIs in the circulation data of the vehicle so they can labelled in the following task. These POIs will then represent the points where the driver stops frequently to, for example, deliver products or refuel the vehicle, making them the key points to establish the vehicle's route.

As such, the found clusters should be able to be plotted in a geographical space, making *latitude* and *longitude* the essential features for this task. Using any more features would produce clusters that could not be accurately translated to a map, making it so that the clustering result would lose its meaning in the business context.

Having this established, the requirements for an algorithm to accomplish this task appropriately should comply with the following criteria:

1. The algorithm should be able to deal with noisy data;
    - Even after the pre-processing done, situations such as traffic jams or sporadic wrongly recorded data should be able to be dealt with, making the process simpler as a result.
2. The algorithm should be able to find an arbitrary number of clusters;
    - The total number of POIs for a vehicle's circulation is unknown.
3. The algorithm should produce a result that is easily interpreted;
    - The classification will work with features created with the results from this task, creating an interest in a easily understood result.
4. The algorithm should have hyper-parameters that are as simple as possible to optimize;
    - The algorithm having easily interpreted hyper-parameters helps streamline a solution that fits different vehicles.
5. The algorithm is scalable with a large amount of samples;
    - Even after resampling the data has a significant amount of samples, specially if the algorithm is ran against data pertaining to multiple weeks or months.
6. New data should be able to be incorporated without having to rerun the algorithm with the data in its entirety;

- There is an interest in incorporating new circulation data in order to find new POIs while maintaining past ones, as new data is produced every day.

7. The clusters found by the algorithm are not constrained to a shape;

- The algorithm to be used should be able to find clusters with non-convex shapes.

Given these conditions, the previously mentioned clustering methods that seem to best fit this situation are density-based methods, where clusters are created by a notion of dense and sparse regions. A parallel can be made to the context of the problem, since POIs are composed by geographical areas where the vehicle stops regularly, creating dense regions. They also comply with most of the presented criteria, specially due to being able to deal with noise and find an arbitrary cluster number. Therefore, they will be given focus. Nevertheless, other algorithms outside of density-based methods will be analysed as a way to establish a comparison. The data used to establish a comparison between all algorithms will use one month of a vehicle's movement, encompassing 32620 samples.

All the run times presented in the following sections are based on a computer with the following specifications:

CPU	Intel Core i5-4200H 2800.0 MHz
RAM	4GB DDR3 SDRAM 800.0 MHz (2)

Table 4.2: Relevant specifications of the computer where the algorithms were ran.

4.2.1 K-means

The first clustering algorithm tried was K-Means, using the scikit-learn[42] implementation. In a general sense, K-Means aims to separate the data into k groups while minimizing within-cluster sum-of-squares[10] of Euclidean distances, in the case of used implementation. In simpler terms, it aims to create k clusters separate from each other as much as possible while maintaining the points contained in each cluster as close as possible.

As the k parameter is user-defined, silhouette score was computed for different k values to try and find a possible number of optimal clusters. The silhouette coefficient tells us how well defined the clusters are by comparing the distance of a sample to other samples in the cluster and the distance of that sample to the nearest cluster that the sample is not a part of[11]. From analysing the plot in figure 4.2, we can then conclude that the optimal value for k is 30 clusters, if we base our analysis around the silhouette score. A number of 10 runs with different centroid seeds was done, being picked the result with the minimum within-cluster sum-of-squares distances, with each run having a maximum number of 300 iterations.

Due to the use of an Euclidean metric to measure distances between points in this algorithm, latitude and longitude were converted from the Geographic coordinate system to the Universal Transverse Mercator(UTM) coordinate system. Both coordinates were then mapped to a two-dimensional space, making it possible to calculate distances between data points accurately.

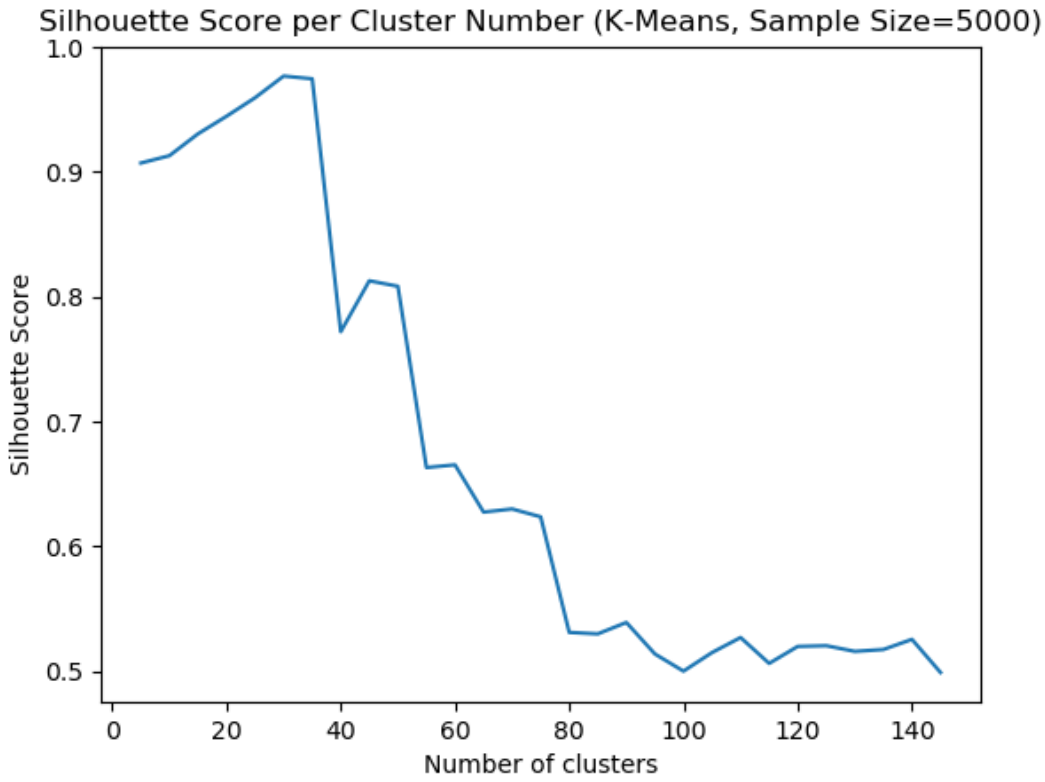

Figure 4.2: Silhouette score variation with different k amount of clusters.

From the analysis of the map output in figure 4.3, we can assess that while the clusters are relatively well grouped due to the optimization of the k value, the clusters obtained do not make much sense with the objective in mind as specially in more urban areas, where the circulation data is more dense, the clusters stretch out over distances that are multiple kilometers long, encompassing to what could be multiple POIs and traffic data, which the algorithm has no capability of filtering out.

Therefore:

1. The data grouping outputs a result that makes sense in context of our problem;
    - Data points are partitioned into clusters even if they do not belong to a POI.
2. Algorithm parameters are intuitive and easy to optimize;
    - The main parameter to optimize, k, is not intuitive in this context as there is no good method to determine the exact amount of POIs outside of using business knowledge, as this number varies completely with each vehicle operation. Fine tuning this parameter is also complex, as increasing or decreasing the k value does not guarantee that better clusters will be found.

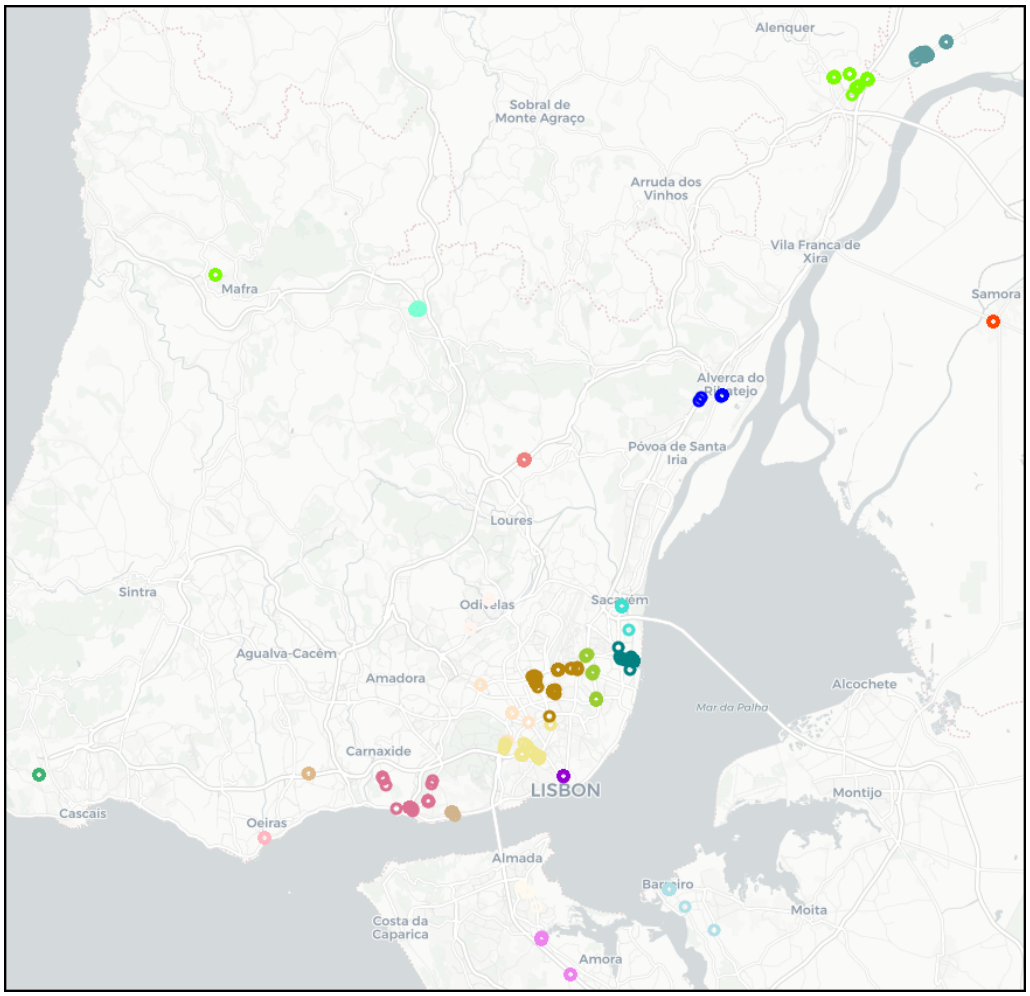

Figure 4.3: K-Means(k=30) output for one month of circulation data. Each same colored circle belongs to a same cluster.

3. Results are stable and are as expected;

    - The results for this circulation data are stable, giving similar results between runs, despite the nature of K-Means where different random starts might lead to different clustering results. Varying the k value also does not produce a result that can be speculated as the data groups in a way that does not correlate with our context and just tries to partition the data to comply with the numbers of clusters the user has defined.

4. Run times are satisfactory;

    - Due to K-Means being a simple and efficient algorithm, run times are extremely fast.

Having these points established, it cannot be said that K-Means constitutes a good solution to this problem, as stable results and fast run times are not enough to compensate for lackluster clustering results.

Cluster Number	Sample Number	Runtime(s)
30	32620	1.31

Table 4.3: Number of clusters found, amount of samples provided and runtime for the K-Means algorithm.

4.2.2 Mean Shift

The next algorithm considered was Mean Shift, also using the scikit-learn[42] implementation, specifically due to not having to specify the number of clusters and having the capabilities to detect noise. Mean Shift works by finding dense regions in the feature space using a probability density function, where these dense regions constitute a local maxima of that function in which a cluster associated to it is delineated[19].

Just like in the case of K-Means, there was an attempt to find the best value for the main parameter of Mean Shift, *bandwidth*, by computing silhouette scores once again. *Bandwidth* represents the resolution of the analysis of the algorithm. Each data point is used as a seed with a maximum number of iterations of 300 and a flat kernel is used.

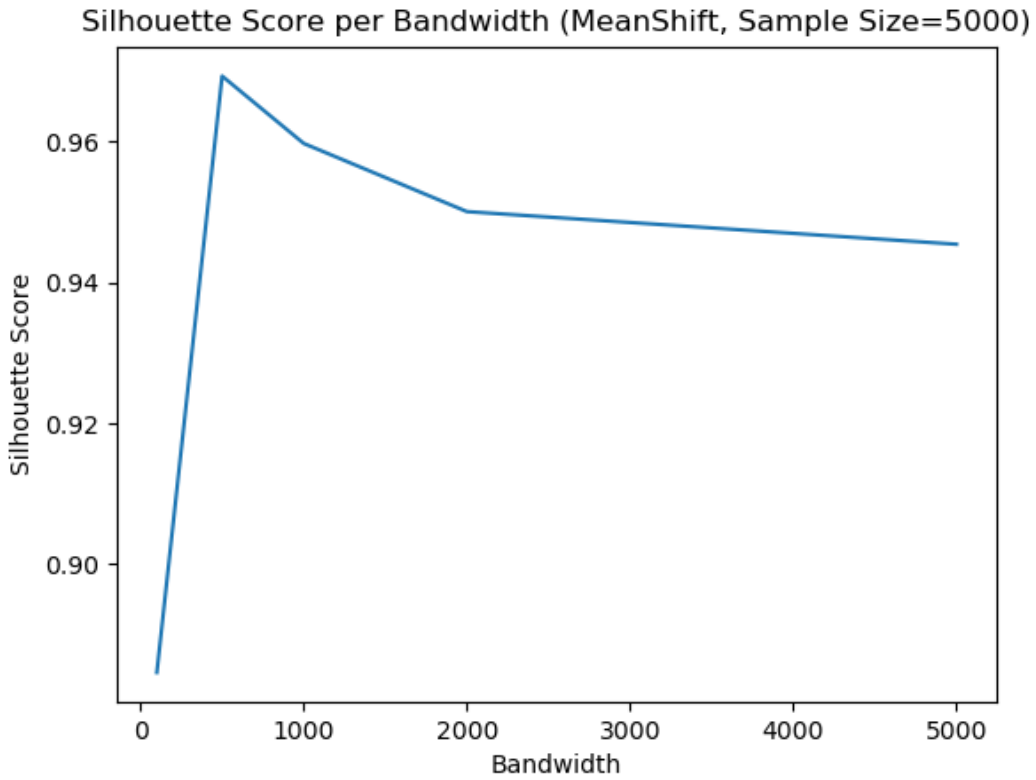

Figure 4.4: Silhouette score variation with different *bandwidth* values.

From the analysis of the plot in figure 4.4, the value to use for the *bandwidth* parameter is 500.

As the feature space preferred is Euclidean[19], coordinates are once again mapped to the UTM coordinate system.

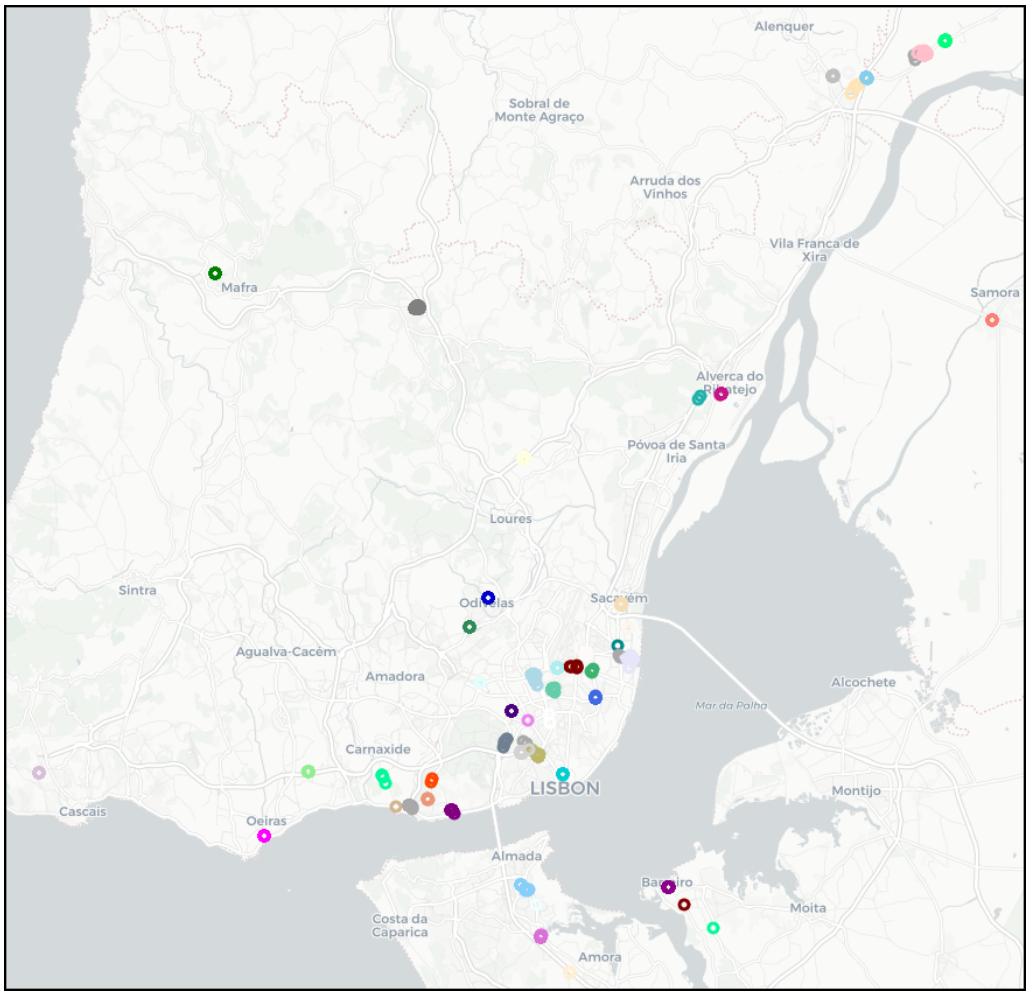

Figure 4.5: Mean Shift(*bandwidth*=500) output for one month of circulation data. Each same colored circle belongs to a same cluster and dark gray colored circles represent noise found in the data.

By analysing the result in figure 4.5, we can notice that the clusters found partition the data a lot more distinctly compared to the K-Means output(Figure 4.3), specially as the number of clusters found were a much higher value too, from 30 as shown in table 4.3 to 60 in table 4.4. While finding a higher number of clusters is not exactly beneficial, as they could be wrongly found, in the context of geographical data where we know that a POI rarely stretches over multiple kilometers, these results seem more promising.

After a better study of the output in figure 4.5, we can also see that the noise detected is correctly distinguished as it constitutes data where the vehicle is stopped in traffic. Nevertheless, only 8 of the samples were detected as noise, meaning the algorithm does not adapt to the context of the data as much as it would be liked, considering coordinates where the vehicle could only be

Cluster Number	Sample Number	Runtime(s)
60	32620	124.35

Table 4.4: Number of clusters found, amount of samples provided and runtime for the Mean Shift algorithm.

in traffic are still grouped into clusters.

Therefore:

1. The data grouping outputs a result that makes sense in context of our problem;
    - A small amount of data points are correctly recognized as not belonging to possible POIs but the amount is almost negligible. As such, while the clusters span over distances that make sense, they also encompass large quantities of wrong data.
2. Algorithm parameters are intuitive and easy to optimize;
    - The main parameter to optimize, *bandwidth*, is hard to optimize due to the little correlation it has with the context of the data.
3. Results are stable and are as expected;
    - The results vary a lot given different *bandwidth* values and it is also hard to notice a pattern in these changes between different *bandwidth* values.
4. Run times are satisfactory;
    - Run times are extremely slow due to Mean Shift algorithms having quadratic complexity on the number of data points and possibly needing multiple iterations to converge[17].

While Mean Shift gave better results compared to K-Means, it has not yet reached a solution that fits our problem satisfactorily, as while the number of clusters is not user-defined and noise is detected, almost all noise goes unnoticed. The exceedingly large run time also makes a solution using this algorithm not feasible, specially as we want to run the algorithm over multiple months or even years of data and not just over a month of circulation data.

4.2.3 DBSCAN

DBSCAN was the next promising algorithm to be experimented with. DBSCAN equiparates clusters to dense regions, where a certain amount of objects have to be in the neighborhood of each other for them to form the clusters, being this amount parameterizable[50].

In the scikit-learn[42] implementation, there are two main parameters to define: *epsilon*, which sets a neighborhood in the format of a radius around a point where it can find another and therefore expand the cluster, and *minimum_samples*, which imposes a minimum number of objects in a neighborhood for them to be considered a cluster.

In this case, choosing the parameters is a different situation from the two cases already presented, as from an initial analysis, it can be noticed that both parameters have a direct correlation with our context: *epsilon* represents how big of a area do we want a POI to expand by with every point included and *minimum_samples* the minimum amount of instances that occur in that area. In the context of our problem this can be translated to a minimum minute count of the vehicle in that area for it to be considered a POI, as after pre-processing, each instance in the data set corresponds to one minute of operation, which was established in subsection 4.1.1, where the observations frequency was lowered from seconds to minutes. As such, parameters were defined by the use of business knowledge, having established *epsilon* with a value of 100, meaning there is a maximum radius of 100 metres to establish the neighborhood around a point and *minimum_samples* with 60, which implies that a vehicle has to remain a non-sequential total time of one hour in an area for it to be considered a POI.

In this case the distance metric used the haversine formula to calculate the distance between each data point so latitude and longitude could be used without converting to another coordinate system, as this formula determines the distance between two geographic coordinates on a sphere.

Cluster Number	Sample Number	Runtime(s)
34	32620	12.53

Table 4.5: Number of clusters found, amount of samples provided and runtime for the DBSCAN algorithm.

From the output in figure 4.6, the first aspect noticed was the noise detected, which in this case is composed by 368 examples, making a complete difference in what clusters are found compared to past results.

With the parameters having direct correlation to our data, we guarantee a result that makes more sense than the previous presented: points where the vehicle stopped due to the traffic are identified as noise, considering a vehicle will rarely stay for multiple minutes in a same point during traffic, being even harder for it to stay for the amount of time designated in *minimum_samples* to end up considered as a POI.

By delimiting the maximum area, we also ensure that the POIs span over a area where clusters can be clearly defined. Setting an *epsilon* too high could make a cluster encompass multiple POIs, while setting it too low could make the algorithm detect multiple POIs where there is only one.

Therefore:

1. The data grouping outputs a result that makes sense in context of our problem;

    - Data is distinctly separated into clusters and noise, creating clearly defined POIs that span over sensible distances and filtering out data pertaining to vehicle stops due to traffic.

2. Algorithm parameters are intuitive and easy to optimize;

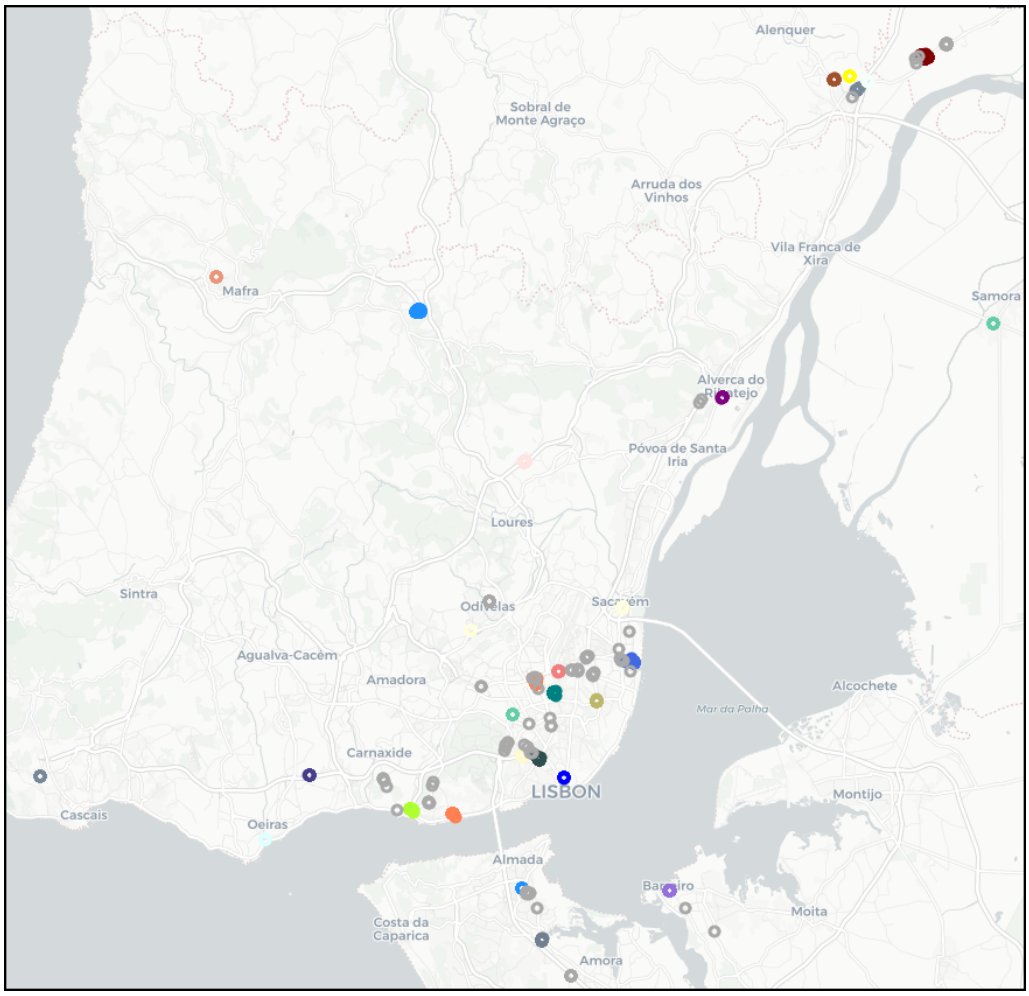

Figure 4.6: DBSCAN(*epsilon*=0.1, *minimum_samples*=60) output for one month of circulation data. Each same colored circle belongs to a same cluster and dark gray colored circles represent noise found in the data.

- The parameters to optimize, *epsilon* and *minimum_samples*, have a direct correlation with the circulation data, making their choice intuitive by the use of business knowledge.

3. Results are stable and are as expected;

   - The results obtained are consistent between executions. Varying the parameters also produces expected results as existing clusters either expand, diminish or disappear and new ones appear, making it easy to detect the pattern in the cluster variance.

4. Run times are satisfactory;

   - Run times evidentiate a great performance of the algorithm.

DBSCAN shows a huge step up compared to the other two algorithms, complying with all the restrictions set for a clustering algorithm in section 4.2. As such, it seems to be a very good solution for our problem, as we expect the algorithm to maintain a good performance even when running it over data for multiple months of a vehicle circulation, as DBSCAN is specifically adapted to be efficient even on large datasets[22].

4.2.4 OPTICS

To try and refine the result obtained by using DBSCAN, the OPTICS algorithm was also considered. OPTICS works as an extended DBSCAN algorithm where *minimum_samples* constitutes the only required parameter. Instead of assigning cluster membership, an ordering of the examples for a *maximum_epsilon* is created[15]. From this information, clusters with varying densities can be extracted.

In the case of the scikit-learn [42] implementation, using a DBSCAN cluster extraction method over the OPTICS output produces the same result as the one presented in the DBSCAN subsection 4.2.3, in the case that *max_epsilon* is equal to DBSCAN's *epsilon*. From the analysis produced by OPTICS, more information can be extracted with the trade-off of a higher computational cost and execution time which can be seen in table 4.6.

Cluster Number	Sample Number	Runtime(s)
34	32620	148.70

Table 4.6: Number of clusters found, amount of samples provided and runtime for the OPTICS algorithm.

Delivery vehicles normally have specific spots to deliver its cargo or to park, meaning that the stop points inside a POI do not vary significantly. As such, the advantages that come from a varying radius to detect POIs are not necessary considering business needs. This along with the higher execution times, that come as a trade-off, make this solution not as desirable, even if a more thorough analysis of the data is produced.

4.3 Establishing a clustering algorithm evaluation measure

Analysing the output from clustering algorithms constitutes an hard process, as it can be troublesome to determine how the data should be partitioned and if those partitions make sense. Fortunately, as the tested algorithms only had the input of two different features, latitude and longitude, the process was able to be evaluated by investigating the output in a interactive map and confirming how the various algorithms behaved. The expected final result of the algorithm is also simple to identify, as the points where the vehicle stop normally represent areas or structures of some significance.

There is an interest in establishing a more objective method to measure the success of an algorithm in clustering the circulation data into a POI, which will increase the support for the conclusions reached in the previous section for each clustering algorithm tested.

Calculating the silhouette coefficient helped creating a better output in the case of K-Means and Mean Shift, where choosing the parameters was not very intuitive, giving a better understanding of how these algorithms partition the data. Nevertheless, the silhouette coefficient cannot be used as performance measure for the algorithms, as this score does not exactly assess how a good cluster is formed from the point of the view of our problem.

With this objective in mind established, for the circulation data used to create the clusters in section 4.2, a list of already known POIs was used to determine if each dataset entry should belong to a POI or not.

By creating a radius around these known POIs, a new feature was added to each dataset example identifying if that entry is located inside a radius of 80 metres around a POI. This new field lets us then determine if a dataset entry should belong to a cluster or not. With this new information determined, we can then establish how well the different algorithms group the data according to our problem. As such, the following outcomes were established:

- True Positives(TP) - this entry should belong to a POI and was assigned to a cluster
- False Positives(FP) - this entry should be considered noise but was assigned to a cluster
- False Negatives(FN) - this entry should belong to a POI but was considered noise
- True Negatives(TN) - this entry should be considered noise and was identified as such

	TP	**FP**	**FN**	**TN**	**Accuracy**
K-Means	26366	6254	0	0	0.808277
Mean Shift	26366	6247	0	7	0.808491
DBSCAN	26318	5934	48	320	0.816616

Table 4.7: Distribution of outcomes and accuracy for clustering algorithms experimented in section 4.2 for the same dataset(Sample Number=32260).

The accuracy values in table 4.7 are calculated using the following formula:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{4.1}$$

K-Means algorithm does not detect any noise, meaning that all data is assigned to a cluster leading to a static accuracy value that is based on the distribution of the data. As such, any algorithm that does not have any noise detecting capability will achieve this same accuracy value and will not be able to improve it under this evaluation measure.

Mean Shift improves the score from the first case minimally due to the negligible amount of correctly detected noise.

Finally, DBSCAN catches correctly a much higher amount of samples as noise which is both reflected in the higher accuracy score and in the visual result presented in figure 2.1. As expected, it obtains the best accuracy between the three options presented.

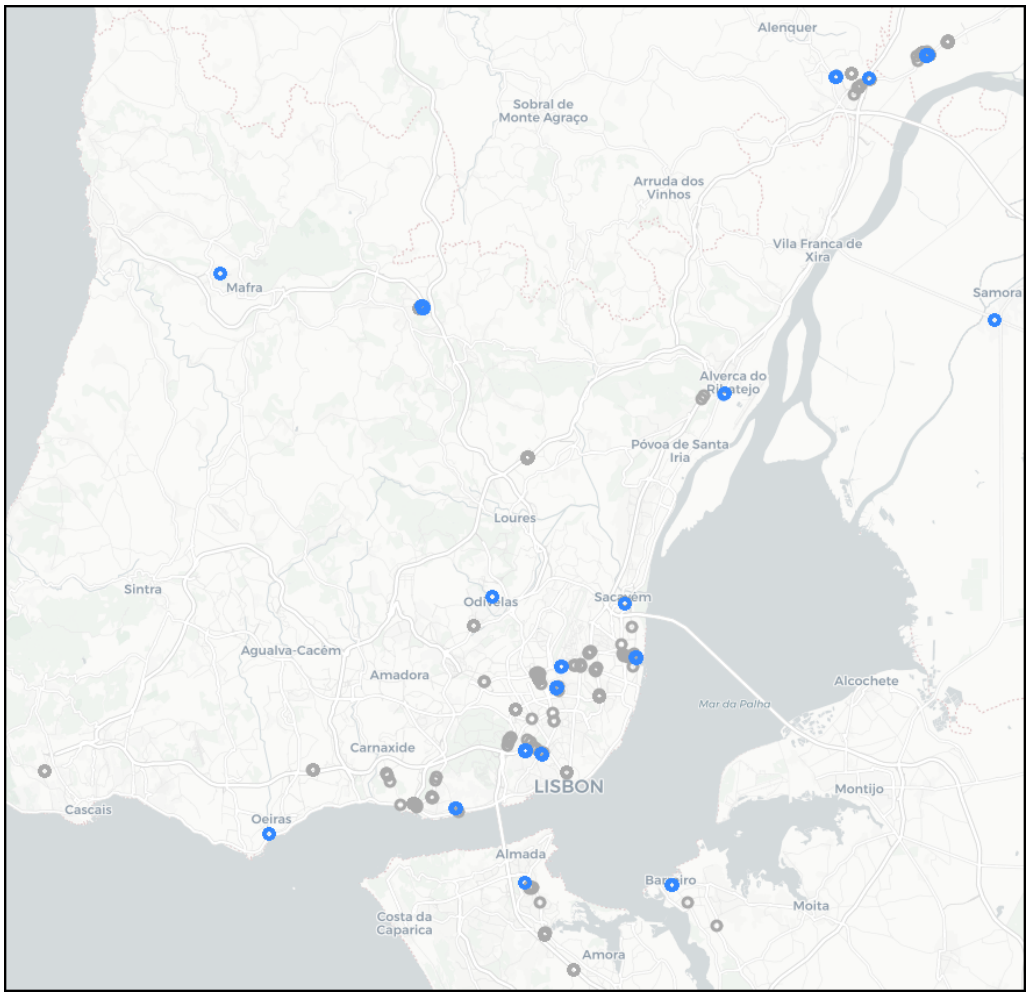

Figure 4.7: Circulation data distribution between POI and noise. Blue points represent data that should clustered by the algorithms to form POIs while dark gray points should be left out of clusters as they constitute noise.

Another meaningful exercise is to plot a map of using this new feature, making it possible to see the visual distribution of noise and data that should belong to a POI. If we compare the output obtained in figure 4.7 to the DBSCAN output in figure 4.6, we identify an equivalence between the blue points, which represent data that should be clustered into POIs, and the colored points in the DBSCAN output, which constitute clusters found.

By further analysis of both outputs, we also observe that in this output some areas that are considered noise, like in the area around Odivelas and Cascais, are identified as clusters in the case of the DBSCAN output. After a closer inspection of these points, using business knowledge, these points can actually be confirmed as a POI, meaning that in the case of DBSCAN, the value

of false positives is not accurate to reality as the algorithm detects POIs beyond the ones already known. In addition to this, another part of false positives detected are not prejudicial to the actual process of finding POIs, as while these data points are considered part of a cluster when they should not, they are grouped due to the high density of points in the area they are in, meaning they will belong to a cluster containing an actual POI.

With these points established, while we cannot say this accuracy score is representative of a real performance value in obtaining all actual existing POIs in the circulation data analysed, it can be used to compare the relative performance between algorithms tested. Another opportunity that comes from the computation of this new feature is the optimization of parameters for the algorithms based on this accuracy value that is directly correlated with our context, which can lead to a better understanding of how parameters should vary with the increase in the amount of data.

4.4 Parameter optimization for the chosen algorithm

As established in subsection 4.2.3, DBSCAN has two main parameters to optimize: *epsilon* representing the maximum radius around a point for it to catch another stops from the vehicle to be integrated in a same cluster and *minimum_samples* which in this context sets the minimum amount of minutes a vehicle has to stay in the area considered, considering each dataset entry represents one minute of operation, as settled in subsection 4.1.1.

By using business knowledge, a sensible value for the distance between stop points can be determined, as the coordinates where the vehicle stops in a same POI do not vary a lot, making the determination of a value for the parameter *epsilon* not complex. However, the same cannot be said for *min_samples*, as the minimum number of minutes that a vehicle has to stay in an area for it to be considered a POI will vary with the amount of circulation data that is being analysed.

As such, using the accuracy measure established in section 4.3, there is an interest in optimizing the *min_samples* value given increasing quantities of data while trying to maximize performance value. In this situation, the performance value will be compared between the same quantities of data, as there is no guarantee of a increase in performance given more data, considering DBSCAN constitutes an unsupervised learning method, meaning that the accuracy could drop if the algorithm does not work as well with newly added data.

The following conditions were then set to determine the best *min_samples* value given increasing data quantities:

- The *min_samples* chosen obtains the best accuracy value for this data quantity.
- With one month of circulation data, at least eighteen clusters have to be detected, being this number determined by the observation of figure 4.7.
- The number of clusters detected between each data increase has to be at least the same or higher.

- The *min_samples* values to be tested for each data quantity starts with 30(thirty minute stay of a vehicle in the cluster area) and increase sequentially using that same value until the conditions set above are met.

Samples	Outlier Nr.	Run Time	Min. Samples	Cluster Nr.	Accuracy
8181	94	3.52844	30	18	0.747097
16399	366	6.95666	60	24	0.809257
24390	442	9.75323	60	26	0.823124
32620	368	13.7619	60	34	0.816616
65371	1109	24.9171	90	42	0.748574
96698	2078	42.039	150	43	0.761857
128374	2510	53.4637	150	47	0.761408
160267	3057	71.565	150	50	0.769784
191786	3221	90.1038	150	54	0.774478
224309	3133	108.165	150	58	0.766737

Table 4.8: Best minimum samples value given increasing circulation data representing one to three weeks and then one to seven months of vehicle movement.

By observing table 4.8, we can notice how a best minimum samples parameter, for the conditions set, varies with different quantities of data. With the initial amount of data, the *min_samples* value has to be increased accordingly to deal with first POIs being found. With vehicle circulation delivery routines being repeated, a threshold for the value of this parameter is reached at three months of circulation data, as the essential POIs belonging to the vehicle's circulation are mostly identified. This is supported by that in first three months of circulation, 43 clusters are found while in the following four months only 15 more are discovered. These 15 new clusters are result from either new delivery points for the vehicle being added or past points being visited enough to be promoted to a point of importance.

With this, a value representing a stop time of two and half hours in a POI is reached for the parameter *min_samples*, which can be used to create the datasets for the classification task in a more accurate way.

4.5 Summary and Conclusions

During this chapter, the clustering process was described. Firstly, data was resampled in order to reduce its size so algorithms can perform faster over a longer period of data. Following that, invalid data was filtered out, making it now possible to apply the clustering algorithms over it. From there, desired constraints for the clustering algorithms were presented along with the analysis of the results of various algorithms. In order to decide with more confidence on the algorithm that will be used to find the POIs, an accuracy value was calculated using known POI data in section 4.3, being these values compared with the visual results obtained in section 4.2. Specifically in the case of DBSCAN, its result was compared with the distribution of data in a known POI dataset,

showing the success of this algorithm in distinguishing between clusters and noise. Finally, a process to optimize the parameters of DBSCAN was established.

A proper decision in an appropriate clustering algorithm is essential, as this step is the cornerstone for the classification task. Label prediction will be performed over datasets created using this algorithm, meaning that a model created using data that was incorrectly generated will make predictions based on wrong data, invalidating the analysis that can be made from that process.

With everything established in this chapter, DBSCAN was then chosen as the clustering algorithm to build into the next task. From the highly intuitive hyper-parameters due to its correlation to our context and results that answer accordingly to algorithm's parameters with relatively low running times, this algorithm presented the best choice between the ones tested.

Chapter 5

Labelling points of interest found in the vehicle's circulation

In this chapter, the process to label POIs is described in detail. To tackle the classification task, feature engineering is done, creating new fields from which the classification algorithms can draw new assessments. The classification setup that is used to analyze the algorithms is explained and the results for the algorithms tested are presented. From the models created using these algorithms, one will present better prediction capabilities, being then used to label the vehicle's stop points.

5.1 Feature Engineering

Feature engineering is an essential step for the success of the classification task, being the objective finding data that the classifier can understand. The features available create a direct impact in the prediction abilities of the algorithm, as data is used as the foundation for model creation.

Many factors in the data can constrain the capability for models to predict, not just the quantity of features but how they are grouped too; well summarized data can lead to simpler models which are faster to create, easier to understand and easier to optimize for.

In this case, two main sources for features were identified: data generated from the aggregation of the fields belonging to examples that compose each cluster and data retrieved from third party sources based on the geographical location of the cluster. Following sections will dig deeper into what kind of features were created or retrieved.

5.1.1 Features generated from the clustering task

The clustering task grouped entries from the dataset generated from raw data, meaning there is an opportunity to create new fields based on the features mentioned in table 4.1 and the notion that each entry represents one minute of operation.

As DBSCAN does not produce clusters with centroids, a medoid is determined instead. A medoid constitutes a representative "central" point of the cluster that exists in the dataset[45], which in this case is determined by calculating the entry that minimizes the total haversine distance

Feature	Description
latitude	Representative latitude value for cluster.
longitude	Representative longitude value for cluster.
min_latitude	Latitude value for the bottom boundary of the cluster.
max_latitude	Latitude value for the top boundary of the cluster.
min_longitude	Longitude value for the left boundary of the cluster.
max_longitude	Longitude value for the right boundary of the cluster.
stop_time	Total time that the vehicle has stopped in the cluster in minutes.
registered_at	Timestamp of creation of the last example that was integrated into the cluster.
vehicle_is_idle	Total time that the vehicle has spent idle in the cluster in minutes.
vehicle_is_stopped	Total time that the vehicle has spent stopped in the cluster in minutes.
refrigeration_diesel_is_on	Total time that the vehicle's diesel refrigeration has been on in the cluster in minutes.
refrigeration_electric_is_on	Total time that the vehicle's electric refrigeration has been on in the cluster in minutes.
refrigeration_electric_is_on_from_battery	Total time that the vehicle's electric refrigeration has been on from the battery in the cluster in minutes.
refrigeration_electric_is_on_from_grid	Total time that the vehicle's electric refrigeration has been on from the grid in the cluster in minutes.

Table 5.1: New features generated from the data grouping performed by the clustering algorithm.

to every other point in the cluster. The latitude and longitude of that data point is then used to represent the cluster.

The field *stop_time* was obtained by counting the number of entries in the cluster, while in the case of *vehicle_is_idle*, *vehicle_is_stopped*, *refrigeration_diesel_is_on*, *refrigeration_electric_is_on*, *refrigeration_electric_is_on_from_battery* and *refrigeration_electric_is_on_from_grid*, the features were obtained by running a sum aggregation over the entries composing the cluster on the fields with the same name from table 4.1, values which were originally formatted as Boolean values.

5.1.2 Features collected from third parties

Using the latitude and longitude characterizing a cluster, new fields can be obtained, in this case by using those coordinates to query outside sources. As such, the Overpass API[12] is used, considering it is a good open source option which allows to retrieve worldwide geographic data from a database copy of OpenStreetMaps[1], with the possible shortcoming of some of the data not being mapped due to built by community effort. By sending a query to this API, custom selected parts of the OSM map data are served, being these queries written in a specific query language, Overpass QL.

Key	Description / Values retrieved
Amenity	Describes useful infrastructures of the community.
	bar, cafe, fast_food, food_court, ice_cream, pub, restaurant, college, kindergarten, music_school, school, university, car_wash, vehicle_inspection, charging_station, fuel, parking, parking_entrance, parking_space, clinic, hospital, pharmacy, social_facility, casino, cinema, community_centre, nightclub, baking_oven, conference_centre,internet_cafe, kitchen, prison, marketplace, vending_machine, shop, loading_dock, post_depot
Building	Describes a man-made structures with different uses.
	apartments, farm, hotel, house, residential, commercial, industrial, manufacture, office, retail, supermarket, warehouse, bakehouse, hospital,greenhouse, garage, garages, parking, service
Craft	Describes places where goods are produced or processed.
	bakery, caterer, hvac, confectionery, electronics_repair, electrician
Highway	Describes the type of roads.
	motorway, trunk, primary, secondary, living_street, pedestrian, service, rest_area
Landuse	Describes the use for the area.
	commercial, construction, industrial, residential, retail, farmyard, greenhouse_horticulture, aquaculture
Office	Describes the type of business that is carried out.
	company, coworking, engineer, logistics
Shop	Describes a place where goods are sold.
	bakery, beverages, butcher, cheese, chocolate, confectionery, convenience, dairy, farm, frozen_food, greengrocer, health_food, ice_cream, pastry, seafoodshopfood, department_store, general, kiosk, mall, supermarket, wholesale, car, car_repair, car_parts, fuel, gas, truck_repair, deli, wholesale
Tourism	Describes places with specific interest to tourists.
	hotel, hostel, motel
Power	Describes places where electric power generation or distribution occurs.
	substation, transformer, generator

Table 5.2: Key:Value pairs retrieved to be used as features by querying OSM.

Firstly, map features[2] to retrieve were identified by determining what structures or areas would make sense to visit given a transport vehicle with a electric refrigeration system. In OSM, map points are tagged using a key and value combination, where keys are used to describe a topic, category or type of feature and values are specific forms of that key. The tags retrieved are listed in table 5.2.

Following that, a query to the API using the format specified in figure 5.1 was built in order to retrieve all the relevant tags listed. Using the coordinates of each cluster, these features were retrieved in a radius of 50 and 100 metres.

By parsing the response received (figure 5.2), the existence of the structures or places described was determined and quantified, creating around 520 new fields. The templates used for feature creation are specified in table 5.3. With this process finished, we had an idea of how the area around a POI was composed, which is essential to the categorization of the point itself, considering there is no guarantee that where the vehicle stops constitutes the POI itself.

Feature Template	**Description**	**Example**
{*key*}:{*value*}_{*distance*}m	Checks if *key:value* structure exists in a radius of *distance* metres.	amenity:fast_food_50m
nr_{*key*}:{*value*}_{*distance*}m	Counts how many *key:value* structures exist in a radius of *distance* metres.	nr_amenity:fast_food_50m
nr_{*key*}_types_{*distance*}m	Counts how many different *value* exist in *key* structures in a radius of *distance* metres.	nr_amenity_types_100m
nr_{*key*}_total_{*distance*}m	Counts the total amount of structures belonging to *key* in a radius of *distance* metres.	nr_amenity_total_100m

Table 5.3: Templates used to create mentioned features using the key:value pairs retrieved.

```
[out:json][timeout:25];
(
  node(around:50, 38.72200,-9.15259)["amenity"];
  way(around:50, 38.72200,-9.15259)["amenity"];
  relation(around:50, 38.72200,-9.15259)["amenity"];
);
(._;>;);
out body;
```

Figure 5.1: Example of a query performed to the Overpass API to retrieve relevant structures around each POI.

```
{
  "type": "node",
  "id": 1586249303,
  "lat": 38.7222244,
  "lon": -9.1524192,
  "tags": {
    "amenity": "cafe",
    "name": "Os cabacinhas"
  }
},
{
  "type": "node",
  "id": 1586249378,
  "lat": 38.7219398,
  "lon": -9.1525238,
  "tags": {
    "amenity": "fast_food",
    "branch": "Rodrigo da Fonseca",
    "brand": "McDonald's",
    "brand:wikidata": "Q38076",
    "brand:wikipedia": "en:McDonald's",
    "cuisine": "burger",
    "internet_access": "wlan",
    "name": "McDonald's",
    "opening_hours": "Mo-Su 08:00-00:00",
    "takeaway": "yes",
    "website": "www.mcdonalds.pt"
  }
},
```


Figure 5.2: Excerpt of a response to a query performed to the Overpass API.

5.2 Labelling vehicle circulation data

With feature engineering done, the final step was then modelling the data by the use of predictive algorithms. Using business knowledge, clusters representing POIs were labelled, reaching five different labels, described in table 5.4, constituting then a multiclass classification problem.

Label	Description
client	Location where a distribution vehicle's goods are delivered.
warehouse	Location where the distribution vehicle's loads its cargo and/or stays in standby.
gas_station	Location where the vehicle refuels.
maintenance	Location where the vehicle's current state is ascertained.
parking	Location where the vehicle's stops temporarily before resuming operations.
other	Non-identified location.

Table 5.4: Possible classes identified in the POIs.

Vehicles working under different contexts require different evaluations. Distinct features are needed to identify the vehicle's operation, depending what is meant to be delivered, for example. As such, to better test the setup created, circulation data for two companies will be considered. Depending on the areas they operate in, even different target classes can be identified. The analysis will expand in the circulation data of one of the companies, concluded by a comparison with the results achieved in the other, so each of the algorithms' performance can come under higher scrutiny. The first company(Company A) operates in Portugal in the delivery to fast food restaurants while the second(Company B) is based in Germany and focuses in the transport to bakeries, meaning that the models created will reach their conclusions using distinctive features in both cases. The geography of Germany also creates the need for an additional label, *parking*. The distances the vehicles have to cover between the POIs is higher when compared to Portugal, making these points of rest important places in the case of the vehicles operating in Germany.

Furthermore, two different datasets for the same circulation data will be compared: one where each entry represents a POI and another where each entry is a stop in that POI. This is done in order to decide if the classification should be performed directly over the result of clustering or if the content of each cluster needs be expanded in order to produce more robust models. In both of these cases, the algorithms will have access to the same fields with the main variation being in the features presented in sub-section 5.1.1, as operation data will be aggregated differently, with the stops dataset constituting a break down of the cluster data.

To choose the fields to use, feature selection is performed on the data to filter the most relevant features for algorithms that do not have ability to do so, enacting an essential step in the labelling process due to the high number of columns created by the data enrichment process. This relevancy is based around mutual information, a measure to quantify the dependence between two variables[43].

With the datasets identified and labelled, various algorithms can be tested using the data generated, producing different models which will have better or worse performances in predicting the labels for each example. This performance will be evaluated using an accuracy metric, as the cost

of a wrong classification is equal: identifying a client as a warehouse or a warehouse as client carries the same risk.

In order to test the robustness of the models created by each algorithm, they have to be trained and tested using differing datasets. As such, k-fold cross-validation is used, being the data split into mutually exclusive k subsets of similar size, where every single data subset is used to test the models created with the rest of the partitions left[32]. The process is then ran a k number of times, where the performance for each model created is measured by the correct predictions over the designated test subsets. By averaging this performance value, we get an idea of how well the models created with each algorithm adapt with different circulation data and if their effectiveness is maintained with unseen data. The number of folds chosen was 10 as it represents an option associated with relatively low bias and variance[32] that will not make run times exceedingly long with the large array of algorithms that will be tested. A cross validated grid search is also performed over the algorithms to fine tune the hyper-parameters of each algorithm, using the evaluation approach already mentioned.

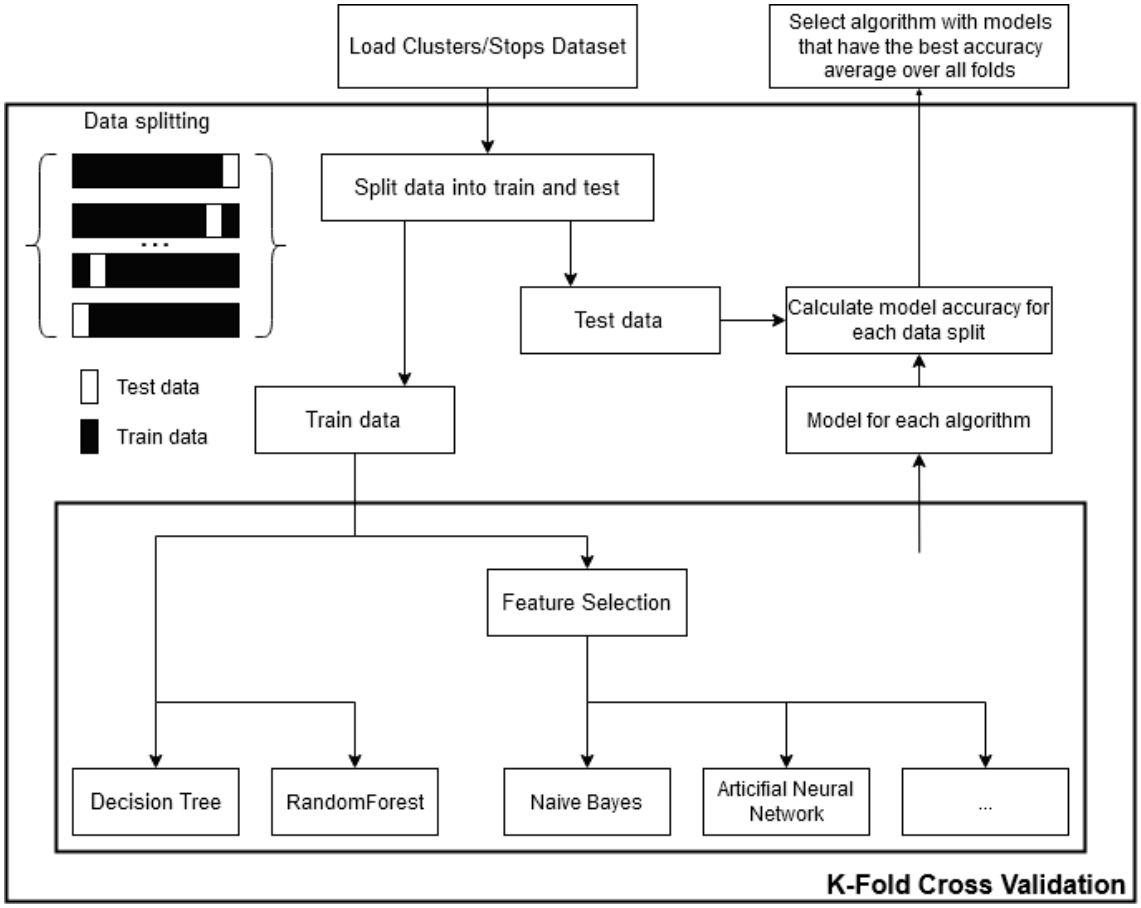

Figure 5.3: Experimental setup for the classification task.

The setup described is summarized in figure 5.3, being the results for each algorithm presented in the following sections. In all cases, sci-kit[42] implementations of the algorithms were used,

except for the instance of XGBoost[4].

5.2.1 Decision Tree

The first classifier tested was a decision tree due to it representing a lightweight solution, with easy to analyse model output. Decision trees are represented by tree-like structures where each non-leaf node encompasses a test on a dataset feature, being each branch the outcome of that process. Each leaf node is a class label, representing a final decision given a path of sequential decisions[32].

In this case, the feature selection capabilities of the algorithm are trusted to pick the correct fields. After running the parameter optimization step, information gain was used to measure the quality of a node's split and a minimum amount of 2 samples to split nodes was set. The maximum depth for the trees created was 4, which was determined by trying to reach a solution that wouldn't lead to the creation of overfitted trees.

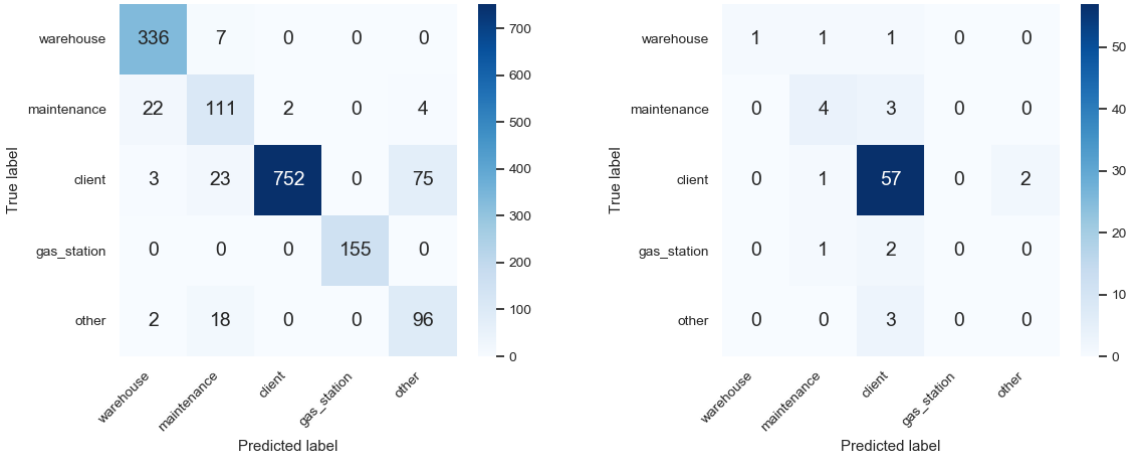

Figure 5.4: Confusion matrix encompassing predictions for the test sets of the ten Decision Tree models generated for the stops (Left, Sample Size=1606, Run Time=0.6 seconds) and clusters (Right, Sample Size=76, Run Time=0.1 seconds) dataset.

The accuracy mean over all folds in the case of stops dataset was 90.3% $\pm$ 2.1% while in the clusters dataset was 81.96% $\pm$ 9.8%. As such, it can quickly concluded that there is a necessity in breaking down each cluster into stops for results with higher accuracy and less variance between models created with different data. The worse results in the clusters dataset can be attributed to the low number of samples with a very skewed class distribution (table 5.5) when compared to the stops dataset. Without relying in a synthetic way to balance the classes, three warehouse POIs can be expanded into 343 times the vehicle has stopped in a warehouse, increasing the representation of this label from around 4% in the clusters dataset to 21% in the stops dataset, for example. Either way, both datasets will be tested with the next algorithms to prove if this balance is consistent in the stops dataset.

True class distribution	warehouse	maintenance	client	gas_station	other
Stops dataset	21.36%	8.66%	53.11%	9.65%	7.22%
Clusters dataset	3.95%	9.21%	78.95%	3.95%	3.95%

Table 5.5: True class distribution in the stops and clusters dataset.

As mentioned before, a decision tree classifier creates a result that can be easily interpreted. As such, an example of a model created with the circulation data being tested can be seen in figure 5.5.

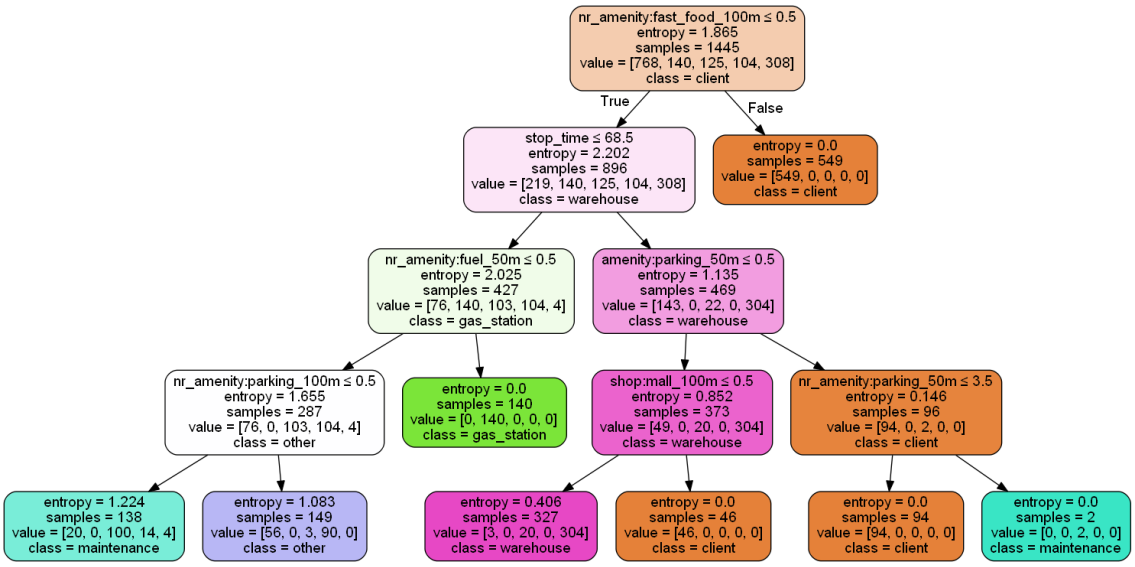

Figure 5.5: Example of a decision tree model created using one of the training folds.

This model highlights right away the main objective of this company in the root node, which is the delivery of edible items to fast food restaurants. After that, when stop times of the vehicle are over an hour, the points where the vehicles stay overnight are identified with the *warehouse* class. Another point of delivery to clients is distinguished with the existence of a mall nearby, as these places normally have fast food joints. Sites where the vehicle refuels are also picked out by the number of gas stations nearby, making indisputable the capabilities of this decision tree classifier in picking out relevant features when there are over 300 available. With this analysis made, the importance of this visualization is emphasized, as we can be more assured of the models created when we know tree structure and the fields picked out to create it make sense in the context of operation of the vehicle. Furthermore, features derived from the vehicle's operation and third parties are being used, showing the importance of the feature engineering step in the whole process.

5.2.2 Naive Bayes

The following analysed classifier was Naive Bayes, as just like in the case of the decision tree, it presents a light and fast option. In this classifier, it is assumed that each of the different attributes produce a independent influence to the determination of a class[32], which cannot be said in the case of the dataset used.

In this case, as mentioned in 5.2, feature selection is performed to select the fifteen most important fields based on its score using mutual information as a measure of field relevance. In this case, the features retrieved were related to vehicle operation timers such as *stop_time*, *vehicle_is_stopped*, *refrigeration_diesel_is_on*, and the presence of amenities connected to the various vehicle's activities like *amenity:fast_food_50m*, *amenity:fuel_50m* or *amenity:parking_50m* and their amount, *nr_amenity:fast_food_50m*.

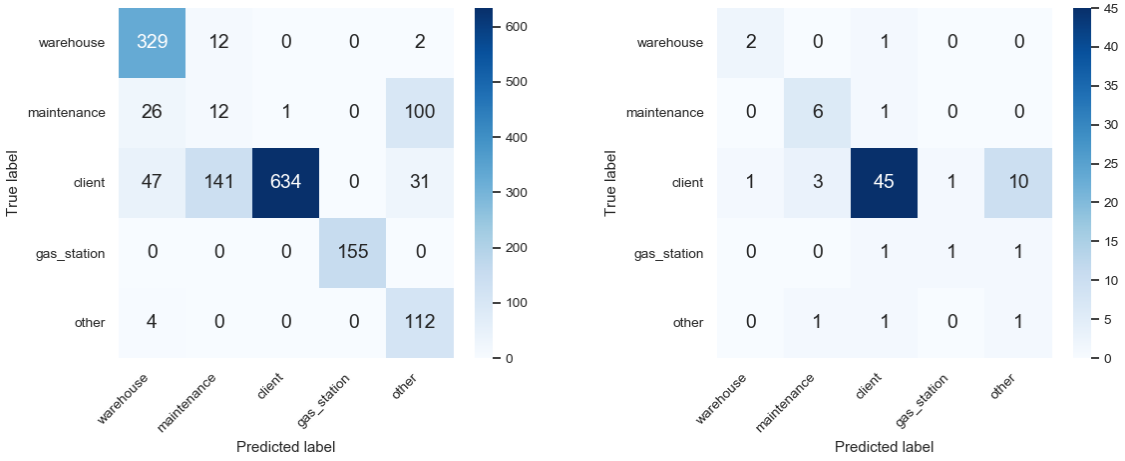

Figure 5.6: Confusion matrix encompassing predictions for the test sets of the ten Naive Bayes models generated for the stops (Left, Sample Size=1606, Run Time=0.3 seconds) and clusters (Right, Sample Size=76, Run Time=0.1 seconds) dataset.

The accuracy mean over all folds in the case of stops dataset was 77.3% $\pm$ 1.9% while in the clusters dataset was 72.9% $\pm$ 11.2%, then reflecting Naive Bayes expectation of independent features.

5.2.3 k-Nearest Neighbors

k-Nearest Neighbors assigns a class to a unknown tuple by analysing the k tuples in the training set most close to it. Each tuple is represented in space by the content of the features that compose it, being the idea of "closeness" defined in this implementation by the Euclidean distance between points[32]. The interest in this algorithm comes from its simplicity, establishing then a baseline result equally like the first two algorithms tested.

Feature selection, as done in subsection 5.2.2, is once again performed, as k-NN has no capabilities to do so and the k-parameter is optimized to k=7, being this value determined by the use

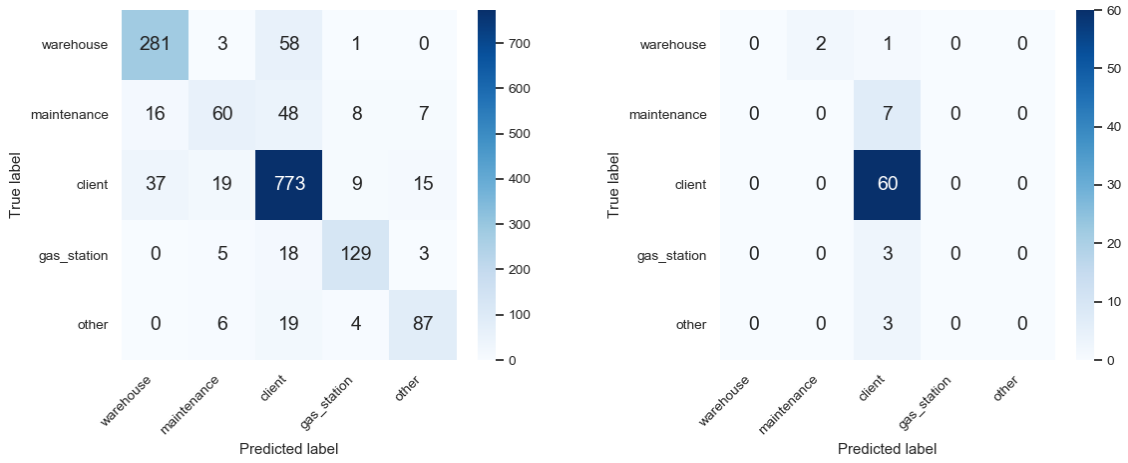

Figure 5.7: Confusion matrix encompassing predictions for the test sets of the ten k-NN models generated for the stops (Left, Sample Size=1606, Run Time=0.3 seconds) and clusters (Right, Sample Size=76, Run Time=0.1 seconds) dataset.

of the already mentioned grid search cross-validation. The distance measure between points used was, as mentioned, an Euclidean distance measure.

The accuracy mean over all folds in the case of stops dataset was 82.8% $\pm$ 2.3% while in the clusters dataset was 79.3% $\pm$ 5.2%.

5.2.4 Support Vector Machines

Due to its very satisfactory results in classification tasks[24], Support Vector Machines were also tested. Training data is mapped into a higher dimension where hyperplanes representing a decision boundary between classes are established, allowing then test data to be identified[32].

After the feature selection is done (subsection 5.2.2), the parameter optimization step led to a C value of 50 and the use of a radial basis function(RDF) kernel.

The accuracy mean over all folds in the case of stops dataset was 67.4% $\pm$ 2.4% while in the clusters dataset was 80.5% $\pm$ 7.9%.

5.2.4.1 Neural Networks

Another possible good option tried were neural networks, where between input and outputs, connections with an associated weight are created, being these connections' weight defined in the training phase as to make class label predictions possible[32].

This classifier is created with the data resulting from the feature selection(section 5.2.2) and configured with one hidden layer with 100 units, using an hyperbolic tan activation function and 500 maximum iterations, as determined by the cross-validation grid search setup.

The accuracy mean over all folds in the case of stops dataset was 95.0% $\pm$ 2.0% while in the clusters dataset was 83.3% $\pm$ 9.7%.

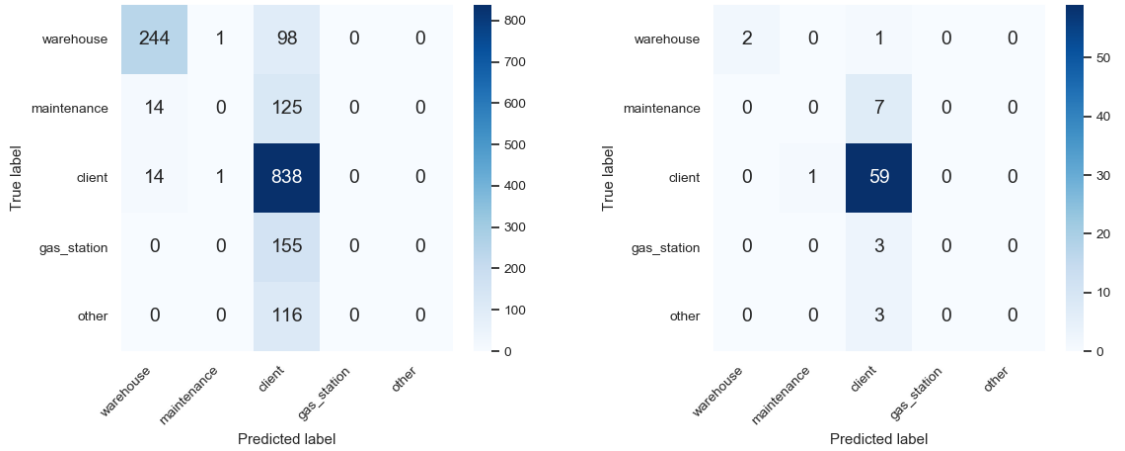

Figure 5.8: Confusion matrix encompassing predictions for the test sets of the ten Support Vector Machine models generated for the stops (Left, Sample Size=1606, Run Time=4.3 seconds) and clusters (Right, Sample Size=76, Run Time=0.1 seconds) dataset.

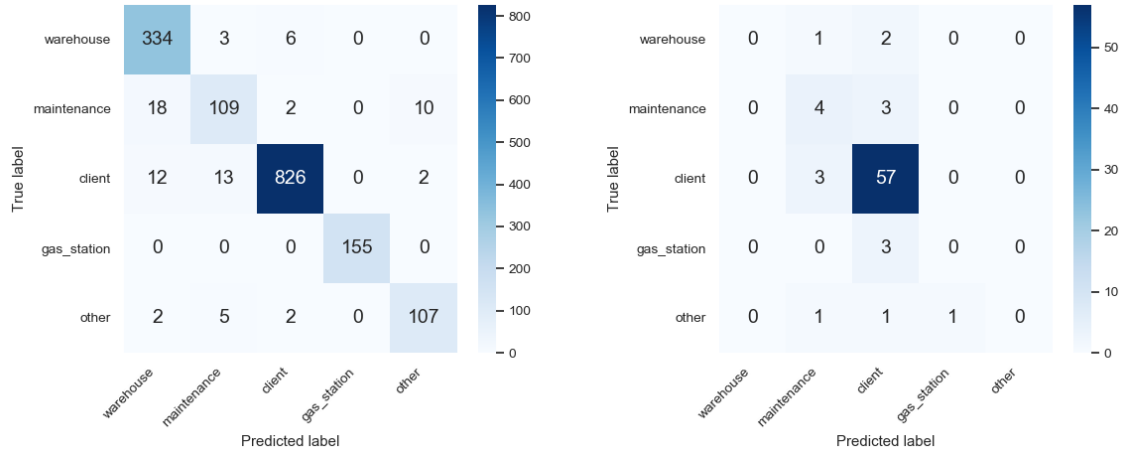

Figure 5.9: Confusion matrix encompassing predictions for the test sets of the ten Neural Net models generated for the stops (Left, Sample Size=1606, Run Time=28.2 seconds) and clusters (Right, Sample Size=76, Run Time=1.9 seconds) dataset.

5.2.5 Random Forests

Given the competitive results obtained by ensemble methods[24], three different types were run, with random forests being the first of those ensembles tested.

In this algorithm, different decision trees with randomized parameters are created using the training data and then these models are used to vote for the popular class in each unseen case[32].

The main parameter to optimize in this case was the number of randomized trees to create which in this case stopped at 1000, as the rest of the parameters were already established from the optimization made for the decision tree classifier, and were therefore reused.

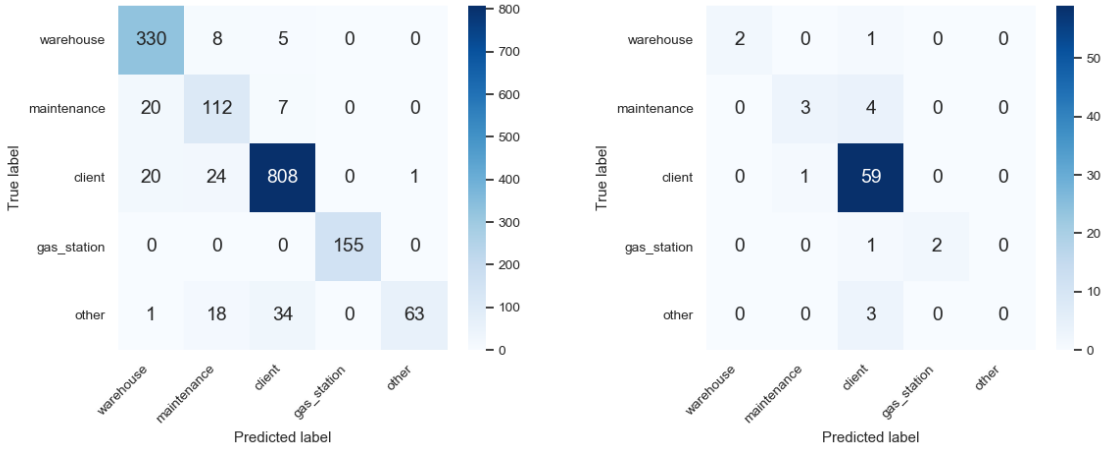

Figure 5.10: Confusion matrix encompassing predictions for the test sets of the ten Random Forest models generated for the stops (Left, Sample Size=1606, Run Time=24.1 seconds) and clusters (Right, Sample Size=76, Run Time=16.3 seconds) dataset.

The accuracy mean over all folds in the case of stops dataset was 91.4% $\pm$ 2.0% while in the clusters dataset was 87.0% $\pm$ 9.7%.

5.2.6 AdaBoost

The second ensemble classifier tried was AdaBoost, in which the focus is further refining a base classifier so as to predict better the harder cases with each iteration[3]. These harder cases will have a higher weight which leads to them be reused as training data, as to prepare the model better in predicting those cases[32].

The classifier that was the target of optimization in this case was the decision tree classifier presented in 5.2.1, meaning that the maximum depth was set to 4 to try not to cause overfitting in the results that this meta classifier will create. A maximum number of estimators to create was set to a 1000.

The accuracy mean over all folds in the case of stops dataset was 97.0% $\pm$ 1.0% while in the clusters dataset was 85.9% $\pm$ 4.9%.

5.2.7 XGBoost

The third and final ensemble method tested was XGBoost, due its recent wide recognition in tackling a multitude of different classification challenges. This algorithm also works around a decision tree ensemble, where in each iteration a new model is created with the intent in optimizing a regularized objective function which assesses the quality of the predictions on the training data and the complexity of the model created, in order to cope with overfitting[18].

The parameters for the trees created have once again a maximum depth of 4 and feature selection, as mentioned in subsection 5.2.2, is also performed, being produced a tree based model.

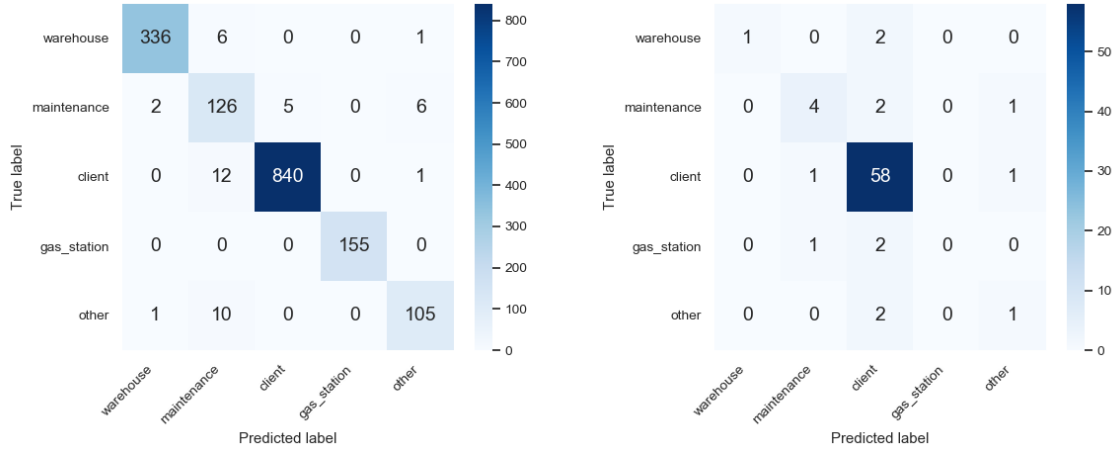

Figure 5.11: Confusion matrix encompassing predictions for the test sets of the ten AdaBoost models generated for the stops (Left, Sample Size=1606, Run Time=457.3 seconds) and clusters (Right, Sample Size=76, Run Time=4.9 seconds) dataset.

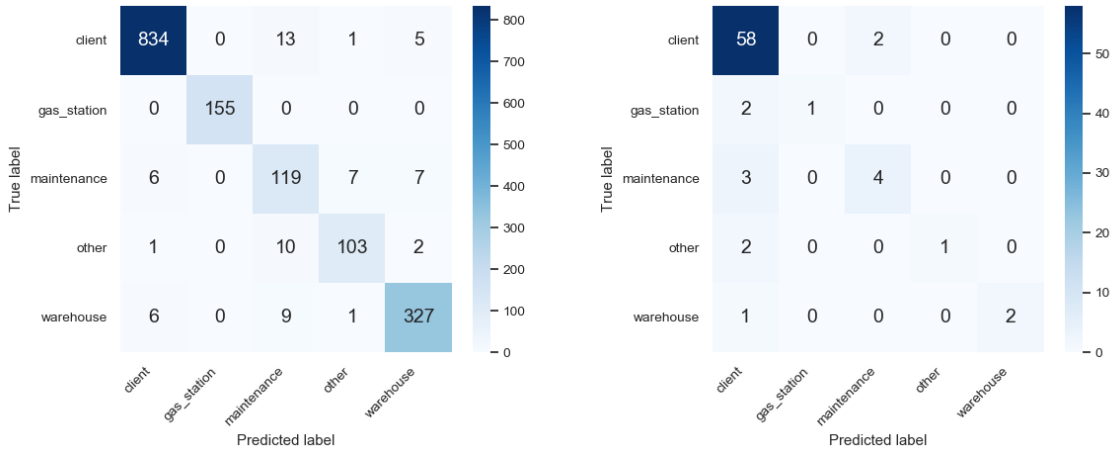

Figure 5.12: Confusion matrix encompassing predictions for the test sets of the ten XGBoost models generated for the stops (Left, Sample Size=1606, Run Time=3.5 seconds) and clusters(Right, Sample Size=76, Run Time=0.6 seconds) dataset.

The accuracy mean over all folds in the case of stops dataset was 95.8% $\pm$ 1.5% while in the clusters dataset was 87.0% $\pm$ 7.9%.

5.3 Summary and Results

In this chapter, a way to classify the clusters found in chapter 4 was presented. With this, new or existing unidentified points where a vehicle stops can be distinguished in the context of its operation, therefore giving meaning to the previous task.

Algorithm	Testing reason
Decision Tree	Lightweight algorithms to serve as baseline classifiers.
Naive Bayes	
k-NN	
Support Vector Machines	Good adaptation to many classification problems.
Neural Networks	
Random Forests	Ensemble methods high accuracy prediction capabilities.
AdaBoost	
XGBoost	

Table 5.6: Testing reason summary for each algorithm.

Firstly, the feature engineering process was described, where new fields were created from the data aggregation with the points composing each cluster and from the collection of geographical data from an external source. Two datasets for the circulation data of a company were presented, in order to determine if the grouping produced from the clusters provided a satisfactory solution or if the data should be broken down to a finer granularity as a POI is composed by the multiple vehicle stops in a point. A setup to test how each classifier would work with unseen data was described and several algorithms were tested using the accuracy metric as a measure, as wrongly predicting a point's label has a same cost independently of the class.

The mains reasons for testing each algorithm are summarized in table 5.6 and the results of the algorithms for each dataset are presented in figure 5.13 and 5.14.

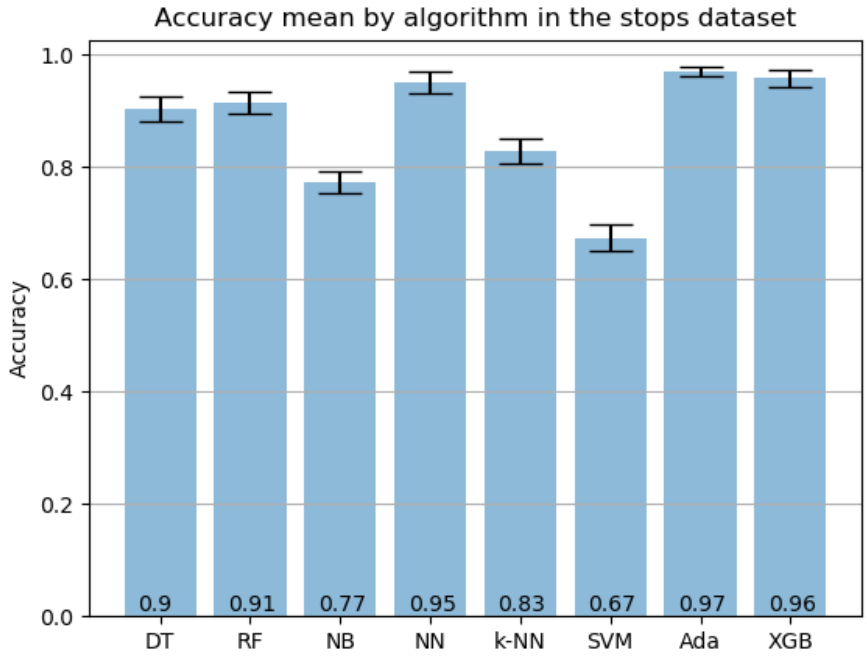

Figure 5.13: Accuracy mean over all models created by each algorithm in the stops dataset.

From the analysis of both graphs, we can notice right away that the clusters dataset produces

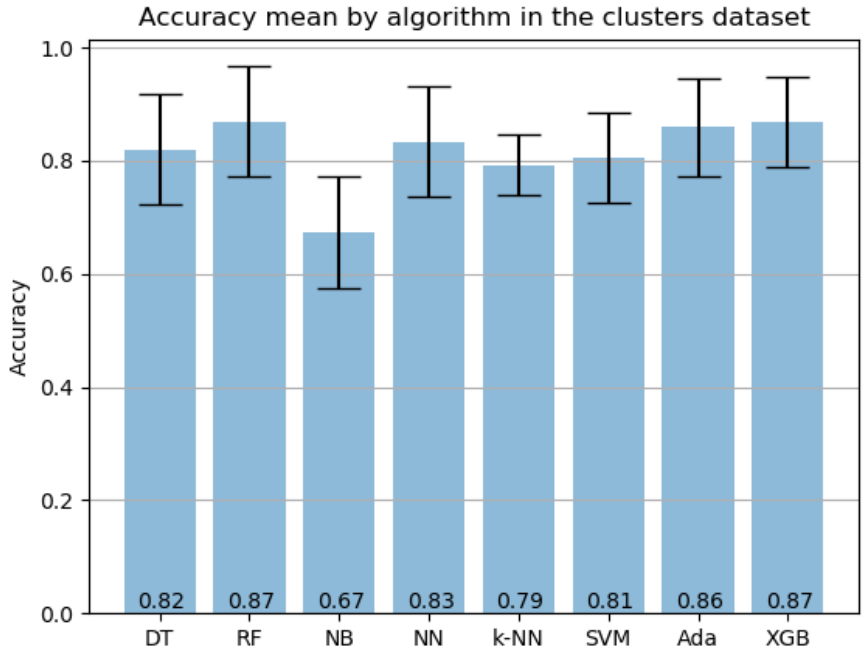

Figure 5.14: Accuracy mean over all models created by each algorithm in the clusters dataset.

worse prediction models except in the case of support vector machines, with also a much bigger variance in predictions capabilities between models created by a same algorithm with different data. As such, it can clearly concluded that for the dataset creation for the classification task, clusters obtained have to be expanded into stops in order to create a dataset with much more samples available that will make possible the creation of models that will achieve higher accuracy results.

In the case of the results in the stops dataset, two ensemble methods, AdaBoost and XGBoost, obtain the higher accuracy results with lower variance along with Neural Networks, being the clear winners in terms of prediction capabilities. Still, between these three algorithms, a big advantage can be definitely noticed in the XGBoost cross validation setup as it only takes 3.5 seconds to run, when compared to the 28.2 seconds of the Neural Nets and the much greater 457.5 seconds in the case of AdaBoost for the same dataset. Another point that should be also remarked is the relatively high accuracy value in the case of the Decision Tree algorithm while being a baseline classifier, specially as the models analysed, like in the case of figure 5.5, make sense in the context of vehicle's operation and are also used as a base estimator in the AdaBoost classifier.

One final analysis to be performed is the comparison of the results presented until now, belonging to a Portuguese company, to ones obtained in the German company mentioned in 5.2, as a way to assess the quality of the results of a same setup in a completely different data context.

In general, the results obtained in both companies are similar, with mostly what seems to be a lower variance between the models created in company B by the k-fold cross validation, except for the case of Naive Bayes, which does not adapt with the data very well in both cases to begin

Algorithm(Stops dataset)	**Company A(Portugal)**	**Company B(Germany)**
Decision Tree	90.3% ± 2.1%	92.1% ± 0.3%
Random Forest	91.4% ± 2.0%	89.1% ± 0.3%
Naive Bayes	77.3% ± 1.9%	76.5% ± 7.9%
Neural Net	95.0% ± 2.0%	94.3% ± 1.1%
k-Nearest Neighbors	82.8% ± 2.3%	90.2% ± 0.7%
Support Vector Machine	67.4% ± 2.4%	86.4% ± 0.6%
AdaBoost	97.0% ± 1.0%	97.9% ± 0.6%
XGBoost	95.8% ± 1.5%	94.6% ± 1.0%

Table 5.7: Algorithm model accuracy comparison between the company operating in Portugal and Germany in the stops dataset.

with.

With these results, there is more assurance that these algorithms can indeed find patterns in data to identify the vehicle stop points in different contexts and will work in predicting the class for unidentified data. While AdaBoost using Decision Trees produces more accurate results, XGBoost should be also an option given the very fast execution times while still keeping a high accuracy. The whole process from raw circulation to identified POIs is then complete, opening up possibilities for application in real life scenarios.

Chapter 6

Conclusions

The work presented established a process to make use of floating car data being retrieved to automatically detect and label the points of interest of a delivery vehicle.

To achieve this, a review on existing work was performed first, where the many uses of circulation data to find and categorize POIs were delimited, specifically also in the case of a commercial fleet. From processed raw circulation data, a setup where multiple clustering algorithms were tried and evaluated was presented, with DBSCAN coming on top due to its very high adaptation to the data's context. Finally, to give meaning to these points found, models created by classification algorithms such as AdaBoost and XGBoost shown a high performance in predicting the objective for the vehicle's stop.

Considering the results obtained, it can be said the whole process adopted was successful in obtaining the desired solution. With a correct input, apart from rare cases, the POIs found by the clustering algorithm always make sense as the chosen algorithm's parameters closely match the business requirements. The detection of these points is based on how long they stay in a certain area, which in turn is also what defines them as a point of importance in the vehicle's operation in the first place.

While the labelling process is not as certain, it still has possibilities to grow and become more robust. As the dataset that behaved best with the classification algorithms is based on the vehicle stops and not in the datapoints themselves, it can expand with more circulation data added, opening up the possibility for the creation of more consistent models.

6.1 Main Contribution

The main contribution from this work is the establishment of a complete process from raw circulation data to labelled POIs, specifically in a context of commercial distribution vehicles. From the literature review realized in the chapter 2, this type of data has not yet received the attention it deserves, possibly due to the difficulty in obtaining it, as there is privacy concerns regarding the identification of companies' operation points. The closest case found[47], presents a similar problem and approach to the solution, but ultimately limits itself to a binary classification problem,

where vehicle stops are described simply as work related or not, while in our case the type of stops is more clearly defined and can vary according to the needs of the data.

6.2 Limitations

The main current limitation of the work presented is the need to create a model for each company. Currently, to properly identify a label such as "Client", the context of the operations of the company have to be taken into account, and, therefore, personalized models have to be created. This leads to the necessity of producing a training dataset for each company, which constitutes a completely manual process, and thus does not scale well with a high amount of companies to analyse.

6.3 Future Work

With the positive outcome shown, the main next focus should be the implementation of the setup described in real life scenarios, like the mapping of POIs for trucks belonging to a company and serving them to fleet managers. With the models created, unidentified POIs can be labelled automatically and statistical information about that point can be presented using the features generated from each entry in the vehicle's operation. This statistical information can range from stop time of the vehicle in the POI to the time the electric refrigeration system was used, opening up the possibilities of calculating other new features like the vehicle's carbon footprint, which nowadays constitutes a important value to keep track of to accompany the increasing environmental policies and newly created low-emission zones. There is also an opportunity in the creation of more complete datasets with the input from fleet managers on wrongly classified points, which would to help the creation of models that predict better.

In the data pre-processing step, as it stands right now, each entry represents a minute of circulation of a vehicle. There is then an interest in performing further aggregation of the data, where the dataset used for the clustering task would have each minute of operation grouped into stops. These stops would hold as much weight as the minutes of operation they contain and would then lower the load on the clustering algorithm without affecting the outcome. The clustering algorithm result would already be in a vehicle's stops format as well, making it usable right away for the classification task.

In both the case of the clusters and stops dataset, it can be noticed that the class distribution is uneven, as shown in table 5.5. As such, methods to cope with class unbalance, such as the creation of synthetic samples, can be tried to determine how they affect the accuracy scores obtained. Also in the classification task, seeing as AdaBoost based on Decision Trees models obtained the best average predicting score, this algorithm could be trained with Neural Nets now being the base estimators. Neural Nets obtained the highest accuracy value outside of ensemble methods, making it a good candidate to produce better predicting models.

This automatic labelling process also sets up new possibilities in the energy management of the electric refrigeration systems, as vehicles moving towards a warehouse, for example, do not need the energy management system to adopt a behaviour as conservative as if it was moving to a client, considering warehouses normally constitute end points for a vehicle's trip.

Appendix A

Data pre-processing graphics

A.1 Latitude

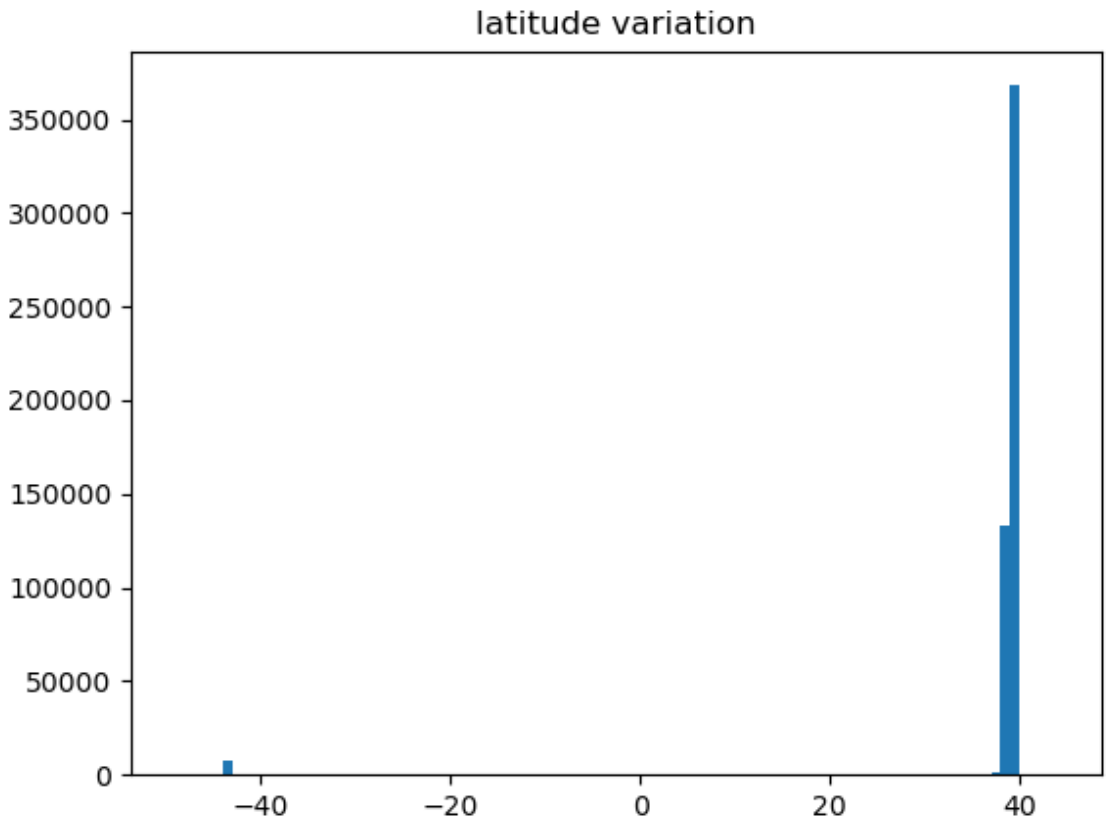

Figure A.1: Field *latitude* value amount variation for a vehicle's circulation data corresponding to one year.

A.2 Longitude

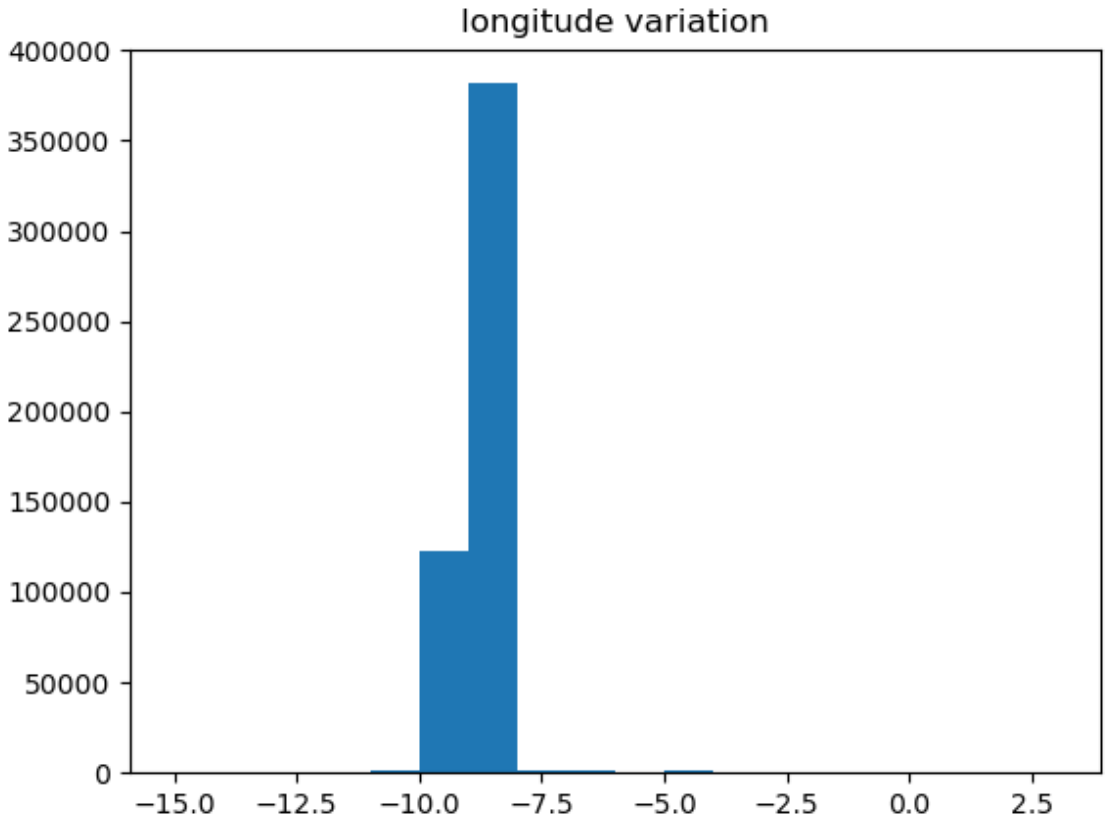

Figure A.2: Field *longitude* value amount variation for a vehicle's circulation data corresponding to one year.

A.3 Revolutions per minute (RPM)

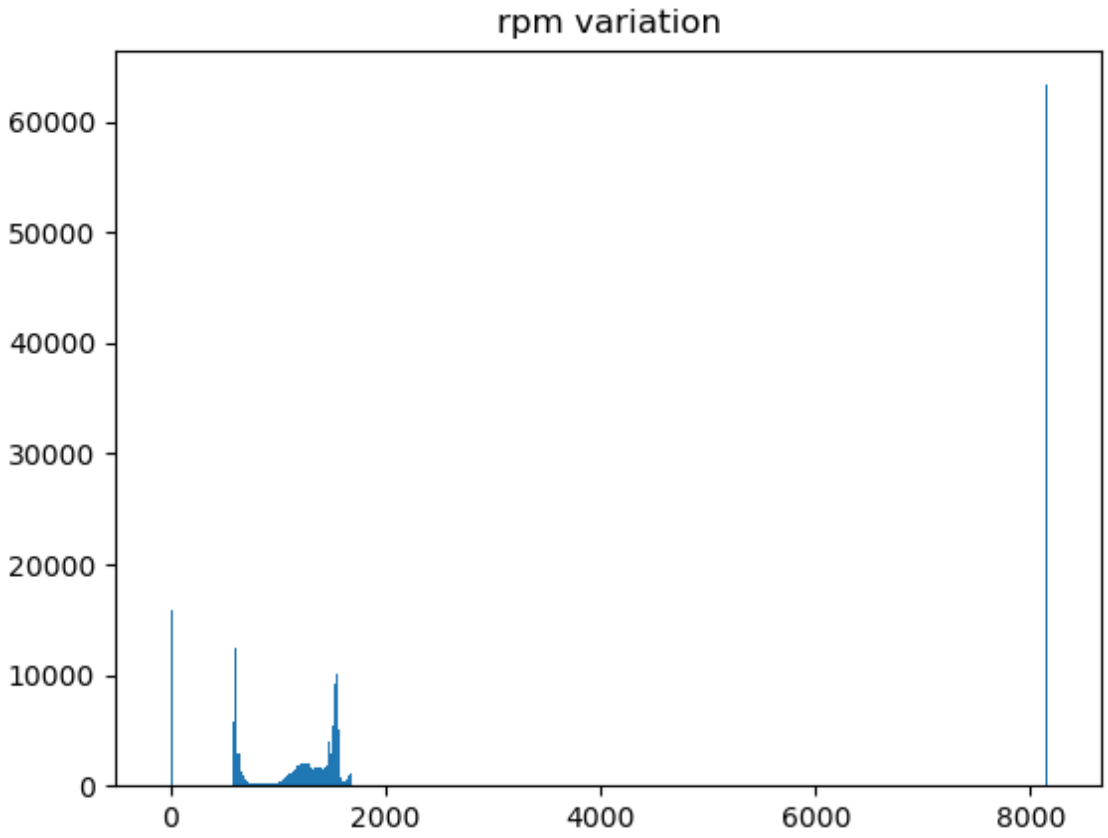

Figure A.3: Field *rpm* value amount variation for a vehicle's circulation data corresponding to one year.

A.4 Vehicle Speed

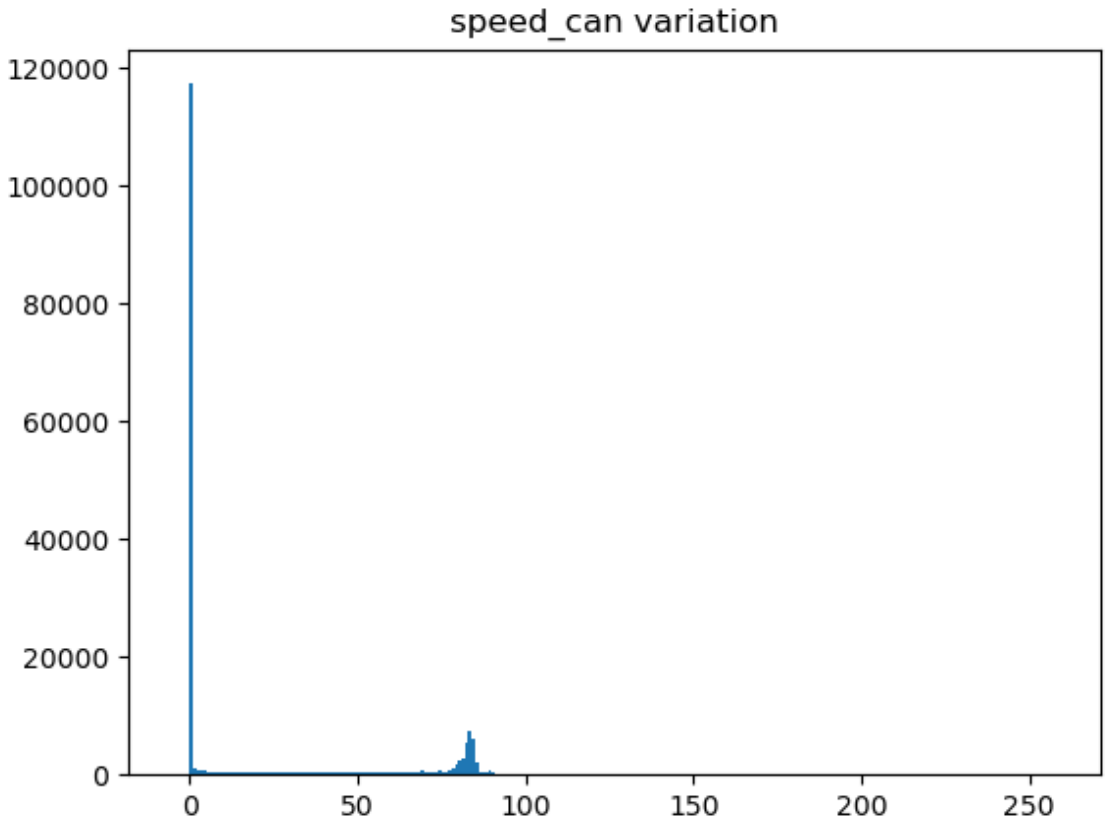

Figure A.4: Field *speed* value amount variation for a vehicle's circulation data corresponding to one year.

References

[1] Openstreetmap. https://www.openstreetmap.org. Accessed: 2020-01-25.

[2] Map features - openstreetmap wiki. https://wiki.openstreetmap.org/wiki/Map_Features. Accessed: 2020-06-04.

[3] Scikit-learn adaboost classifier. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html. Accessed: 2020-06-17.

[4] Xgboost. https://xgboost.ai/. Accessed: 2020-06-18.

[5] Google maps. https://www.google.com/maps. Accessed: 2020-01-25.

[6] Low emission zones. https://urbanaccessregulations.eu/low-emission-zones-main/. Accessed: 2020-02-05.

[7] Google earth. https://www.google.com/earth/. Accessed: 2020-01-30.

[8] Postgresql. https://www.postgresql.org/. Accessed: 2020-05-18.

[9] Accuracy of gps data. https://wiki.openstreetmap.org/wiki/Accuracy_of_GPS_data. Accessed: 2020-05-19.

[10] Scikit-learn k-means. https://scikit-learn.org/stable/modules/clustering.html#k-means. Accessed: 2020-05-20.

[11] Scikit-learn silhouette score. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html. Accessed: 2020-05-20.

[12] Overpass api/overpass ql. https://wiki.openstreetmap.org/wiki/Overpass_API/Overpass_QL. Accessed: 2020-06-04.

[13] European Global Navigation Satellite Systems Agency. Gnss market report, October 2010. https://www.gsa.europa.eu/sites/default/files/GSAGNSSMarketreport2010.pdf.

[14] Charu C Aggarwal. *Data mining: the textbook*. Springer, 2015.

[15] Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: ordering points to identify the clustering structure. *ACM Sigmod record*, 28(2):49–60, 1999.

[16] Juan P Caraça-Valente and Ignacio López-Chavarrías. Discovering similar patterns in time series. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 497–505, 2000.

[17] Miguel A Carreira-Perpinán. A review of mean-shift algorithms for clustering. *arXiv preprint arXiv:1503.00687*, 2015.

[18] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[19] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.

[20] Corrado De Fabritiis, Roberto Ragona, and Gaetano Valenti. Traffic estimation and prediction based on real time floating car data. In *2008 11th international IEEE conference on intelligent transportation systems*, pages 197–203. IEEE, 2008.

[21] Philippe Esling and Carlos Agon. Time-series data mining. *ACM Computing Surveys (CSUR)*, 45(1):1–34, 2012.

[22] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[23] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37–37, 1996.

[24] Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *The journal of machine learning research*, 15(1):3133–3181, 2014.

[25] Tak-chung Fu. A review on time series data mining. *Engineering Applications of Artificial Intelligence*, 24(1):164–181, 2011.

[26] Tak-chung Fu, FL Chung, Vincent Ng, and Robert Luk. Pattern discovery from stock time series using self-organizing maps. In *Workshop Notes of KDD2001 Workshop on Temporal Data Mining*, pages 26–29, 2001.

[27] Pierre Geurts. Pattern extraction for time series classification. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 115–127. Springer, 2001.

[28] Kevin Gingerich, Hanna Maoh, and William Anderson. Classifying the purpose of stopped truck events: An application of entropy to gps data. *Transportation Research Part C: Emerging Technologies*, 64:17–27, 2016.

[29] Lei Gong, Hitomi Sato, Toshiyuki Yamamoto, Tomio Miwa, and Takayuki Morikawa. Identification of activity stop locations in gps trajectories by density-based clustering method combined with support vector machines. *Journal of Modern Transportation*, 23(3):202–213, 2015.

[30] Terry Griffin, Yan Huang, and Ranette Halverson. Computerized trip classification of gps data extended abstract. *Relation*, 10:9220, 2008.

[31] James D Hamilton. *Time series analysis*, volume 2. Princeton New Jersey, 1994.

[32] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[33] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[34] Maarten Houbraken, Steven Logghe, Marco Schreuder, Pieter Audenaert, Didier Colle, and Mario Pickavet. Automated incident detection using real-time floating car data. *Journal of Advanced Transportation*, 2017, 2017.

[35] Rob J Hyndman and George Athanasopoulos. Time series patterns, April 2018. https://otexts.com/fpp2/tspatterns.html.

[36] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[37] Eamonn Keogh, Jessica Lin, Sang-Hee Lee, and Helga Van Herle. Finding the most unusual time series subsequence: algorithms and applications. *Knowledge and Information Systems*, 11(1):1–27, 2007.

[38] Eamonn J Keogh and Michael J Pazzani. An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In *Kdd*, volume 98, pages 239–243, 1998.

[39] Grainne Kerr, Heather J Ruskin, Martin Crane, and Padraig Doolan. Techniques for clustering gene expression data. *Computers in biology and medicine*, 38(3):283–293, 2008.

[40] Oded Maimon and Lior Rokach. *Data mining and knowledge discovery handbook*. Springer, 2005.

[41] Lara Montini, Nadine Rieser-Schüssler, Andreas Horni, and Kay W Axhausen. Trip purpose identification from gps tracks. *Transportation Research Record*, 2405(1):16–23, 2014.

[42] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[43] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence*, 27(8):1226–1238, 2005.

[44] Mahmood Rahmani, Erik Jenelius, and Haris N Koutsopoulos. Route travel time estimation using low-frequency floating car data. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pages 2292–2297. IEEE, 2013.

[45] Chotirat Ann Ralanamahatana, Jessica Lin, Dimitrios Gunopulos, Eamonn Keogh, Michail Vlachos, and Gautam Das. Mining time series data. In *Data mining and knowledge discovery handbook*, pages 1069–1103. Springer, 2005.

[46] Maribel Yasmina Santos and Adriano Moreira. Automatic classification of location contexts with decision trees. 2006.

[47] Leonardo Sarti, Luca Bravi, Francesco Sambo, Leonardo Taccari, Matteo Simoncini, Samuele Salti, and Alessandro Lori. Stop purpose classification from gps data of commercial vehicle fleets. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 280–287. IEEE, 2017.

[48] Li Wei and Eamonn Keogh. Semi-supervised time series classification. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 748–753, 2006.

[49] Xiaopeng Xi, Eamonn Keogh, Christian Shelton, Li Wei, and Chotirat Ann Ratanamahatana. Fast time series classification using numerosity reduction. In *Proceedings of the 23rd international conference on Machine learning*, pages 1033–1040, 2006.

[50] Rui Xu and Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on neural networks*, 16(3):645–678, 2005.

[51] Shi Zhong, Taghi M Khoshgoftaar, and Naeem Seliya. Clustering-based network intrusion detection. *International Journal of reliability, Quality and safety Engineering*, 14(02):169–187, 2007.

[52] Yang Zhou, Chao Yang, and Rongrong Zhu. Identifying trip ends from raw gps data with a hybrid spatio-temporal clustering algorithm and random forest model: a case study in shanghai. *Transportation Planning and Technology*, 42(8):739–756, 2019.