Parking Availability Prediction Using Traffic Data Services

Klandev Ivan[1][0000−0002−5412−8902], Tolevska Marta[2][0000−0002−0344−6564], Mishev Kostadin[2][0000−0003−3982−3330], and Trajanov Dimitar[2][0000−0002−3105−6010]

[1] IT Labs, Skopje 1000, Macedonia
klandev.ivan@students.finki.ukim.mk
[2] Faculty of Computer Science and Engineering, Skopje 1000, N.Macedonia
marta.tolevska@students.finki.ukim.mk,
{kostadin.mishev,dimitar.trajanov}@finki.ukim.mk

Abstract. Implementation of a smart parking system providing predictions about real-time parking occupancy is considered to be crucial when managing limited parking resources. In this study, we present a methodology based on machine-learning regression models for predicting parking availability. We use traffic congestion information and garage occupancy as input to the model gathered from public services, and we predict the parking availability in the same garage sixty minutes later. When using the XGBoost regression model, we achieve MSE=0.0567 which confirms the efficiency of our methodology. Additionally, we find that the timestamp and the current parking availability value are the most influencing factors in prediction which proves the auto-regressive nature of the observed problem.

Keywords: Public parking · Parking prediction · Smart city · Smart parking · Traffic congestion · Garage availability · Regressive model · Machine learning

1 Introduction

In recent years the number of everyday car users has grown rapidly. However, the capacity of roads and parking garages has always lagged behind the growth of motor vehicles, so finding an available parking spot is becoming a much more difficult problem in almost all countries. Research by Yanxu Zheng, et al. [24] says that 30% of the traffic jams in cities are caused by drivers that are searching for a free parking slot. Since drivers do not have any real-time information about expected parking occupancy, they cannot predict the right place where they can find an available parking slot [7]. Ineffective management of parking lots and their parking slots has a big affection on this issue [23]. Besides the increase of traffic and lack of available parking slots, this leads to more road accidents, and air pollution increase as well [20]. Another survey says that nowadays, almost 55% of the world's population lives in cities, and that number is expected to rise

to 68% by the year 2050 [1, 3]. All of this information demonstrates the necessity of implementing a smart parking system.

Thanks to the evolution of IoT (Internet of Things), there is a way to monitor and provide real-time information about parking occupancy and available parking slots [18]. However, the problem here is that not all of the public parking lots and garages are monitored and keep track of its parking occupancy [9]. Therefore, the system will not have a source from where it can get the appropriate availability information in real-time. To easily solve this problem, some predictions based on data collected from the past will have to be made [10]. This approach is useful not only for the private vehicle owners [4] that want to organize their trips while traveling, but for the transportation companies as well [14]. For example, agencies like city traffic and planning departments could use the information about predicted traffic flow, and parking occupancy to speed up the process of planning and managing the transportation [13]. By having a visualized preview of collected textual data about public parking occupancy anyone can easily find the places where a prediction system needs to be implemented [11]. During the last few years there have been various types of researches, which propose systems that provide information in real-time [13].

In this paper, we propose an approach for predicting the available parking slots near the driver's desired destination. This is achieved by training a model based upon data about parking occupancy and traffic data from past parking events. As a first step, a review of the literature and analyzes of existing and proposed parking access systems was performed. Real-time traffic data services were analyzed and appropriate applications that collect data on parking availability and traffic flow were built. This data was collected in a 4-minute interval in a period of 6 months. Because making predictions based on real-time information about parking occupancy will not be accurate enough, having data about similar parking events for the previous 6 months, will make the predictions far more precise and reliable. The next step is to process and merge the collected data. We built a regressive model based on the Extreme Gradient Boosting (XGBoost) regressor which implements the concepts of Gradient Boosting and for fine-tuning the hyper-parameters of the proposed model, we implemented the grid search technique. The approach and the methodology used in this paper show great efficiency and give promising results. This is confirmed by achieving mean squared error of 0.0567.

1.1 Related Work

Predicting and analyzing the availability of parking spaces is a broad area of research where there are many opportunities to use different technologies and methodologies. There have been many scientific types of research and applications in this area over the past few years. Existing practices, proposed solutions, and applications in the field of predicting the availability of parking spaces will be analyzed. Some of the most popular prediction systems, are based on the assumption that the vehicles are arriving at the parking slots in garages following a Poison distribution [12]. These types of systems predict the occupancy using

Markov Chain [17], but the problem here is that they are not considering different factors like time of day, type of day, month, weather conditions, etc, which can affect the prediction's accuracy. According to the way of obtaining parking data, the proposed solutions and systems can be divided into two categories:

- Systems and proposed solutions based on data from parking infrastructure. Where the required data is obtained from sensors, video cameras, or other types of devices that are placed in the parking spaces themselves or the garages.
- Systems and proposed solutions based on user data - Crowdsourcing. Where data is obtained from users, most often via mobile phones or cars.

Systems and proposed solutions based on data from parking infrastructure In this type of system and proposed solutions, the data is obtained from sensors, video cameras, RFID, or other types of devices that are mounted in the parking garage, parking area, or in the parking lot itself. These systems work in such a way that sensors or cameras detect whether a parking space is occupied or not. This information is sent to a central server where all parking spaces in the garage are stored. Users often get information about the availability of parking spaces through appropriate mobile applications. In addition to information on availability, some applications offer the option of booking and paying for parking, as well as navigating to the booked parking space or to the garage where parking spaces are available. The advantage of this type of system is that parking information is very accurate and available in real-time. The main disadvantage of these systems is the cost of installing the devices (sensors, video cameras, etc.), also there are additional costs for connecting the equipment with different interfaces, as well as maintenance costs.

The primary target of the proposed parking system from Yogesh Tayade and M.D. Patil [21] is to find the nearest available parking space for the user who drives a car in a particular area, to reserve a parking space, and to provide him with navigation instructions to the parking place or garage . For this system, an Arduino-based interface has been developed to communicate with sensor devices, sending and processing sensor data to the central unit. Another proposed solution is the system by Snehal Shinde et al. [19] and it is derived from the idea of the IoT. The system uses a WSN (Wireless Sensor Network) consisting of an RFID (Radio-frequency identification) reader that calculates the percentage of free parking spaces in each parking space. Luca Mainetti's proposed system and solution [15], among others, is an automated parking management system based on multiple technologies, such as RFID4, WSN3, NFC, and smartphones. The system collects information on the availability of free parking spaces and then uses a mobile application to direct drivers to the nearest free parking space. The system, proposed by Daniel Becker and others, is a system for monitoring parking garages using fixed video cameras and LiDAR scanners [2]. Many other systems use similar or the same types of technologies and devices.

Systems and Proposed Solutions Based on User Data - Crowdsourcing Systems and proposed solutions based on user data - crowdsourcing are systems where data is obtained from users, mostly through their mobile phones or through devices in cars such as video cameras or ultrasonic sensors. There are two ways to collect data: explicit data collection where free parking spaces are explicitly reported by users through appropriate mobile applications, and implicit data collection where data is sent from applications on mobile phones or cars without direct user interaction.

Crowdsourcing based systems are widespread and their main advantage is the low operating costs. It is not necessary to install devices in parking garages or parking spaces, in most cases, it is simple enough to make a mobile application. However, the main disadvantage of this type of system is the need for a large number of users to generate enough data to provide reliable information about parking and the quality of information. It is not guaranteed that free parking spaces will stay free until the users get to them and park their vehicles. One of the main challenges of crowdsourcing applications is how to motivate users to use the application, to collect relevant data. The system proposed by Martin Margreiter et al. [16] is a field study of free parking spaces, using cars equipped with ultrasonic sensors. This system uses crowd-sourced data from built-in car sensors to detect and predict the availability of parking on city streets. ParkMaster [8] is a system that uses cell phone sensors to help drivers find a parking space in urban areas. The ParkMaster system consists of an Android application that sends camera and sensor data to a cloud server where data is processed and stored in parking spaces. Similar to ParkMaster, iParking [22] is a system where parking access is recognized by analyzing images obtained from videos on smartphones or the Android iParking app. After processing them, if a driver requests a location for a free parking space, the iParking system displays the nearest free parking spaces near the driver's location with the option to navigate to the appropriate parking space.

All of these proposed solutions are getting the data either by crowd-sourcing or by cameras and scanners in garages. We made a web crawler that provides us with the data by collecting it from web sites of garages that have implemented an occupancy monitoring system. Our main goal is to get the data from those parking garages, and then use it to make predictions in real-time, about expected parking availability in parking garages that do not have an occupancy meter.

2 Analysis of Real-time Traffic Information Services

Various services offer real-time traffic information. Analysis of these services was made due to the need to collect traffic data which will then be used for parking analytics and predictions of parking availability.

2.1 Here WeGo

Here WeGo is a web mapping service developed by HERE Global B.V. which allows users to display maps and satellite images of the streets. It also provides

detailed navigation when users do not have an active internet connection on their mobile devices. The user only has to enter the address to the desired destination and Here WeGo calculates the directions, distance, and time required to reach that destination. Real-time traffic data is also included in the calculations. The maps can be downloaded locally on a smartphone using the Here WeGo application for mobile devices and then the same maps can be used without an active internet connection which is quite useful when traveling abroad and having limited internet access.

Like other services, Here WeGo has an SDK and API for developing third-party applications that will use Here WeGo services. The tool development kit is divided into four products:

Java Script API - contains many libraries and methods for processing and visualizing maps, displaying navigation directions, real-time traffic, and more.

Rest APIs - using HTTP GET methods, maps, real-time traffic data, navigation directions, data is returned in XML or JSON format.

Mobile SDKs - Allows the use of Here WeGo services and maps in applications designed for Android and iOS

Automotive - A basic software package for the development of embedded (in vehicles) and mobile navigation solutions

Traffic API - an integral part of Rest APIs which provides real-time traffic information and traffic incidents for a specific area through REST services.

2.2 Comparison of Real-time Traffic Services

To choose which of the analyzed real-time traffic services will be used, a comparison will be made between them, according to the following already established characteristics: Area coverage; Getting the data and its accuracy; Traffic data format.

Area coverage Bing Maps and HERE WeGo real-time traffic services have almost the same coverage, with the largest coverage being the United States and Europe. Unlike them, Google Maps has more coverage and offers traffic and smaller road information.

Getting the data and its accuracy Bing Maps receives real-time traffic data from HERE WeGo. Here WeGo received traffic data from several sophisticated sources such as car sensors, road sensors, and data from traffic control and control centers. Google Maps receives real-time traffic data from Android mobile devices, whereby calculating the speed of users along the road, Google Maps can generate maps of the current traffic situation.

Traffic data format The real-time traffic data format that Bing Maps and Google Maps restore from their services is a traffic layer that can only be added to their maps and a layer from which traffic can only be visually analyzed. Unlike Bing Maps and Google Maps, HERE WeGo, in addition to providing a traffic layer for its map, also has a REST API where real-time traffic data is obtained in JSON format that can be easily processed and analyzed.

Table 1. Data structure of the garages in Madison

Feature of real-time traffic services	**Description**
ID	Garage identifier
Name	Garage name
URL	Link to more detailed garage information
vacant_stalls	Number of free parking spaces in garage

We chose HERE WeGo as the best real-time traffic service. HERE WeGo's biggest advantage over Bing Maps and Google Maps is the data format we get from the services. HERE WeGo through its REST API allows us to obtain data in JSON format that can be easily analyzed software. The next chapter will explain in more detail the format of traffic data obtained from HERE WeGo.

3 Sources and Structure of Data on Parking Garages

For this paper, it was necessary to analyze data from parking garages, ie how the number of vacancies during the day for several months. Several sources and ways of obtaining such data were analyzed, in the end, it was decided to collect this type of data through the Internet, ie from parking garages that have websites that provide real-time information for the number of free parking spaces. That is, the data should be collected by a web crawler. This decision was made because parking operators rarely allow access to their historical data. Garages we found that have real-time data on their websites are in these cities in the USA: Pittsburgh, Madison, Ashville, Ann Arbor, Seattle, Santa Barbara, Santa Monica, and also Winchester in the UK.

Each of these cities has a different number of garages and a different format for its data. To show the format of this data, we will use one of the cities as an example, ie the city of Madison, USA. Madison's garages data is in JSON format and their structure is shown in Table 1 while the information about each of the garages is shown in Table 2. The number of garages in this city, that we collected data for, is 6.

Table 2. Information about each garage in Madison

ID	**Name**	**vacant_stalls**
2_1	Overture Center Garage	650
2_2	State Street Capitol Garage	625
2_3	Government East Garage	500
2_5	State Street Campus Garage	950
2_6	Capitol Square North Garage	560
2_9	Brayton Lot	245

4 Collection and Processing the Data Related to Traffic and Parking Garages

The data related to the traffic and parking garages were collected for six months starting from 01.10.2017 to 31.05.2018. Once the data was collected, the next step was to process it so that it could be used to train machine learning models. JAVA desktop application called Data Collector was created to collect that data. The application at a set time interval, which is set at startup, takes data on traffic and parking garages and stores them in an appropriate CSV file. In the beginning, in addition to the time interval, the type of data to be collected is defined. That is traffic data or data on parking garages. In addition, for the processing of data related to traffic and parking lots, a JAVA desktop application called Data Processor has been created, by which the collected data is processed and stored in an appropriate CSV file. This application also accumulates and stores traffic data and parking garages in an appropriate CSV file.

4.1 Description of the Data Collector Application

The Data Collector application is a JAVA application that collects data on current garage occupancy and current traffic near the garage. When starting the application, the {data_type} argument is defined, which defines the type of data to be collected, ie if it is 1, then data on the occupancy of the garages are collected, and if it is 2, data on the traffic around the garages are collected.

Collection of traffic data around garages The Data Collector application makes calls to HERE WeGo's Traffic API every 4 minutes. The time is 4 minutes because the free use of the Traffic API allows up to 100,000 calls and transactions per month, ie: 24 hours x 60 minutes = 1440 minutes in one day, 1440 minutes : 4 = 360 calls per day for one city, 360 daily calls x 8 cities = 2880 calls per day for all 8 cities, 2880 x 30 days = 86 400 calls per month for all cities at 4 minute intervals. The Data Collector application makes a call to the Traffic API for each city individually wherein the call we define the area for which we want to get real-time traffic data. Traffic API allows multiple configurations to define the area for which we want to get traffic data. The configurations supported by the Traffic API are described in Chapter 2.3. The Data Collector application uses the proximity configuration of an area where the area we define is a circular area defined by the center of the circle, latitude and longitude, and radius in meters. For each city, circular areas that cover all the garages in the city are defined. For instance, values of the proximity configuration for the city of Madison: *City Name* - Madison; *Center of the area* (latitude, longitude) - (43.073183, -89.389554); *Radius* - 2400.

The data obtained from the Traffic API is stored in a CSV file for each city and day separately. The CSV file contains the following data: *Date* - Date and time when the data was obtained, in the appropriate timezone of the city; *CityId* - City Identifier; *TrafficInfo* - JSON structure from Traffic API. The JSON structure is described in detail in Chapter 3.

Collection of data on garage occupation The Data Collector application makes calls to the appropriate URLs for garage occupations for each of the eight cities, every minute. The resulting data is stored in a CSV file for each city and day separately. The CSV file contains the following data: *Date* - Date and time when the data was obtained, in the appropriate timezone of the city; *LotId* - Garage Identifier; *CityId*- City Identifier; *LotName* - Garage name; *FreeSpaces* - Number of free parking spaces.

4.2 Description of the Data Processor Application

The Data Processor application is a JAVA FX desktop application. Its purpose is to process the traffic data that is in JSON format and to combine the processed data on the traffic flow and the occupancy of the parking garages, in one CSV file for each garage separately.

Processing the traffic data Traffic data is collected with the Data Collector application as described in Chapter 4.1 and stored in JSON format, ie as received from the Traffic API without being further processed. Using the Data Processor application, the processing of a CSV file with traffic data is done as follows:

1. From the CSV file, take the date, time, and JSON structure obtained from the Traffic API and start parsing.
2. All parking garages for the city are taken and for each of the garages the traffic flow is calculated divided into three different radiuses (100m, 200m, 300m), Fig. 1:
3. Then, for each radius are calculated: Traffic flow in the direction from the radius boundary to the parking lot - IN and Traffic flow in the direction from the parking lot to the radius limit - OUT

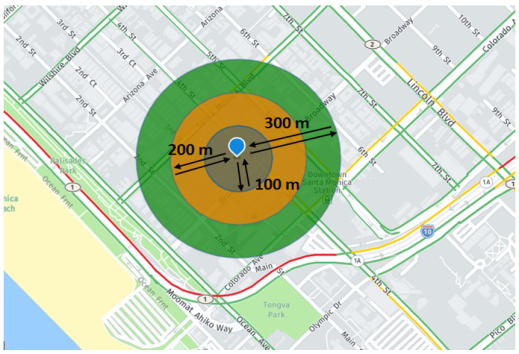

Fig. 1. Traffic flow calculation for three different radiuses

The traffic flow for a radius and parking garage is calculated as follows:

1. Take all FI (Flow Item) elements from JSON. The FI element includes traffic flow information on part of the roadway.
2. For each FI element, its geo-coordinates are taken through the SHP element, which is a series of pairs of latitude and longitude arranged in the direction of movement of a given part of the road.
3. Check the radiuses to which the FI element belongs. To belong to a given radius, it is sufficient for one element of the sequence of geo-coordinates of the SHP element to be inside the circle defined by the radius, and geolocation of the garage as the center.
4. Once it is determined that FI belongs to a certain radius, it is next to determine whether the flow is IN or OUT, so that if the geolocation indicating the beginning of FI is closer to the garage than the geolocation indicating the end of FI, then the flow is IN otherwise it is OUT.
5. After determining the type of flow (IN or OUT), the calculation of the traffic flow is the sum of all JF (Jam Factor - factor of traffic congestion on a given part of the road, presented in a range of 0.0 to 10.0) values that are defined in the CF (Current Flow) element for each FI.

Merging traffic flow data and parking garage occupations After processing the traffic flow data and storing them in the appropriate CSV files, the next step is to merge the traffic flow data and the garage occupancy into one file for each garage individually. The data merger is done with the Data Processor application. Firstly, for each garage the application takes 2 CSV files, one is for traffic flow and the other is for garage occupancy. Later, for each date, time slots are created, starting at 00:00 and increasing by 4 minutes. The time slot is 4 minutes because the Traffic API is called for every 4 minutes. Once the relevant traffic flow data for the appropriate time slot has been found, it is necessary to add data on the occupancy of the garages. Because garage occupancy data were collected every minute, an additional calculation was made here, taking into account the average occupancy of the garage on the records found between the two-time slots. It also calculates the type of day in the week (Monday, Tuesday...) and the number of free parking spaces for the garage is calculated in percentages.

5 Dataset Creation and Description

This study uses parking garages availability and traffic information. Both of them are combined to create a single comprehensive dataset containing all important information about the availability of parking places for the garage and the information about the traffic around the garage. The merged dataset contains the following variables:

- *TimeSlot* - Time slot of the record (time slots are divided into 4 min)
- *DayType* - Type of day where the following values are possible: 1 - Monday, 2 - Tuesday, 3 - Wednesday, 4 - Thursday, 5 - Friday, 6 - Saturday, 7 - Sunday.

– *FreeParkingSpaces* - Free parking spaces expressed in percentages (0.0-1.0)
– *JamInR1, JamInR2, JamInR3* - Factor of traffic congestion in radius 1, radius 2 and radius 3, respectively, with direction to the garage;
– *JamOutR1, JamOutR2, JamOutR3* - Factor of traffic congestion in radius 1, radius 2 and radius 3, respectively, with direction from the garage;

6 Machine-learning Model

This section presents the research methodology which we use to develop the regression models for parking availability detection. Figure 2 presents the steps included in the methodology which we present in the following subsections.

6.1 Input and output Variables

We were collecting data about parking garages occupancy and the traffic flow near them for a period of 6 months, in a 4 minute interval. In total we have gathered data from 92 garages in 8 different cities across USA and UK. 80% of that data is used in the training phase, while the rest 20% is used in the validation phase.

We use the merged dataset in order to extract 33 features that we provide as an input to the regression model. The extracted features are described in Table 3. The model predicts the target variable y which presents the parking availability 60 minutes after the timestamp of the input variable $y0$.

6.2 Regression Model

We build a regressive model based on Extreme Gradient Boosting (XGBoost) regressor [6, 5] which implements the concepts of Gradient Boosting. We choose the XGBoost model because it controls over-fitting and it outperforms the other regressive models in many Kaggle competitions. XGBoost uses an ensemble of weak predictors in order to achieve better results.

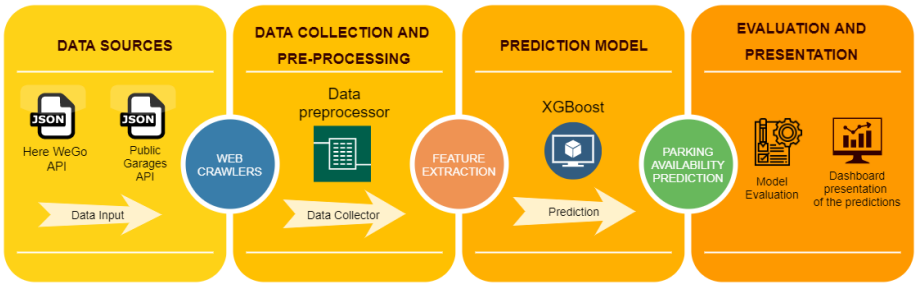

Fig. 2. Methodology

Table 3. List of features used as input to the regression model

Name	Description	Range
H	$H = h + m/60$ where h is the hour and m is the minute of the record.	$[0 - 24.00)$
Day_Of_Week	Nominal variable which presents the day of the week.	[0-7]
$y0$	The current parking availability in a garage.	$[0.0 - 1.0]$
y0_{i}	A real number value which presents a lagged information about parking availability i*4 min before.	$[0.0 - 1.0]$
avg_y0_{i}	Statistical measure which presents the average value of $y0$ in the sequence of records i minutes before the timestamp of y_0.	$[0.0 - 1.0]$
std_y0_{i}	Statistical measure which presents the standard deviation of $y0$ in the sequence of records i minutes before before the timestamp of y_0.	$[0.0 - 1.0]$
TotalJamINRadius_{i}	Factor of traffic congestion in radius i with direction to the garage	$[0.0 - 1.0]$
TotalJamOUTRadius_{i}	Factor of traffic congestion in radius i with direction from the garage	$[0.0 - 1.0]$

In our study, we use the XGBoost model for parking availability prediction problems. We input the presented input features extracted from the initial dataset. We predict the output variable y which presents the parking availability. The evaluation is performed on a dataset which contains garages positioned on different locations.

To fine-tune the hyper-parameters of the proposed model, the grid search technique is implemented. The grid search is an exhaustive search through a manually specified subset of the parameters' space of the model. The grid search is guided by the mean absolute error metric. We find that the XGBoost model achieves the best results when the number of estimators is 1000, learning rate 0.1, and max-depth 3.

6.3 Metrics

The performance and the method's accuracy were studied in terms of mean squared error (MSE) and mean absolute error (MAE). We used these metrics to examine the validity of the model, Table 4.

In statistics, the mean squared error (MSE) measures the average squared difference between the estimated values and the actual value. MSE is a risk function, corresponding to the expected value of the squared error loss. While mean absolute error (MAE), in statistics, is a measure of errors between paired

Table 4. Mean squared and mean absolute error summed for all garages in each city

City Name	Mean Squared Error	Mean Absolute Error
Pittsburgh	0.0565 ± 0.018	0.02924 ± 0.0097
Madison	0.0484 ± 0.0064	0.0260 ± 0.0043
Asheville	0.0582 ± 0.0049	0.0312 ± 0.0022
Ann Arbor	0.058 ± 0.016	0.0313 ± 0.0114
Seattle	0.0576 ± 0.0289	0.0294 ± 0.0177
Santa Barbara	0.0556 ± 0.0117	0.0313 ± 0.0094
Winchester	0.0565 ± 0.0314	0.0292 ± 0.0199
Santa Monica	0.0495 ± 0.0130	0.0244 ± 0.007

observations expressing the same phenomenon. The mean absolute error is a common measure of forecast error in time series analysis.

$$MSE = (\frac{1}{n})\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \quad \text{and} \quad MAE = (\frac{1}{n})\sum_{i=1}^{n}|y_i - x_i|$$

7 Results

Using the XGBoost algorithm, predictions that show the expected number of free parking slots in the next hour, for a specific area, were made Fig. 3, Fig. 4.

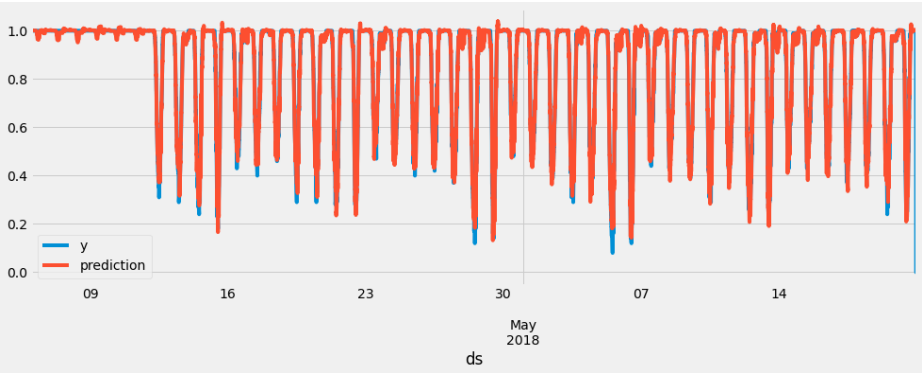

Fig. 3. Prediction of the parking availability for the "The Brooks" garage in Winchester (mean squared error of 4.03% and mean absolute error of 1.55%)

The predictions are based on data from previous parking events and previous traffic flow data. Figure 5 shows the different importance of each of the features that the predictions are based upon. The figure proves that the time component included in the feature set influences the most the accuracy of the model. It means that the number of free parking lots is tightly coupled to the hour during

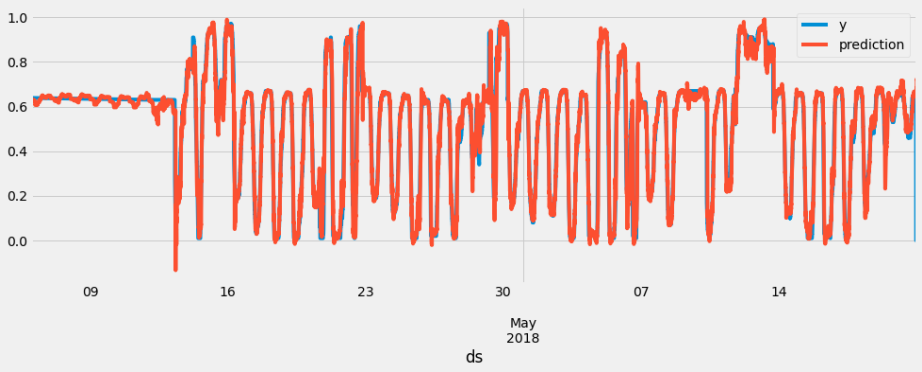

Fig. 4. Prediction of the parking availability for the "Convention Centre" garage in Pittsburgh (mean squared error of 5.92% and mean absolute error of 2.98%)

the day (H) and the day of the week (DAY_OF_WEEK) and it shows the similarity between observations as a function of the time lag between them. The previous value of the available parking lots, y_0, is the second influential variable which improves the model's accuracy and it demonstrates the auto-regressive nature of our problem.

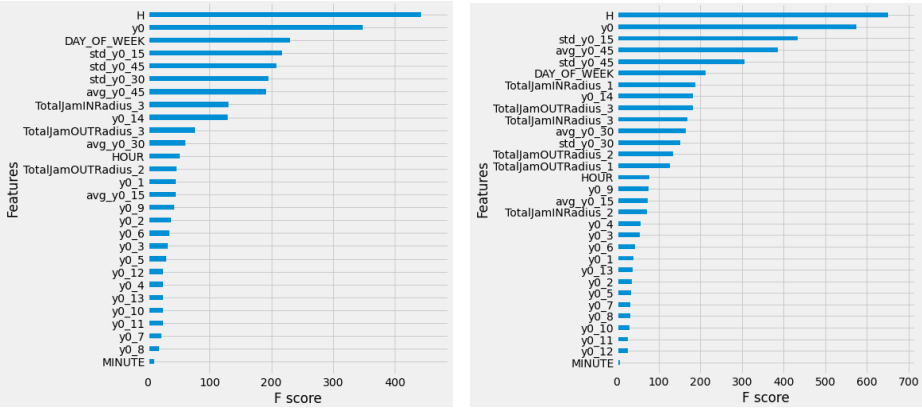

Fig. 5. Feature importance for "The Brooks" garage in Winchester (left) and "Convention Centre" garage in Pittsburgh (right)

8 Conclusion

Nowadays, the number of drivers in cities is rising constantly, so the time needed to find an available parking space is rising as well. To speed up this search time

and reduce traffic jams and air pollution, implementation of a smart system is required. Regarding this problem we provide an alternative solution to the ones previously developed. Our system will provide real-time information about parking availability, by making predictions according to previously collected data about the number of parking events and traffic flow as well. By using these predictions, information about parking availability will be provided even for garages that have not implemented a parking meter. This solution is based on four steps, collecting data about parking occupancy and traffic flow, merging this data, processing it, and then making predictions based on it. To predict the number of expected available parking slots, the XGBoost algorithm was used and showed great accuracy during the training and testing phase. Additionally, we find that the timestamp and the current information for parking availability are the most influencing variables in prediction. It proves that the parking prediction problem can be viewed as an auto-regressive problem.

As future work, we will leverage the gathered dataset and we will additionally evaluate recurrent neural networks (LSTM/GRU). After that, to show the predicted data a web dashboard will be built.

References

1. Bakıcı, T., Almirall, E., Wareham, J.: A smart city initiative: the case of barcelona. Journal of the knowledge economy **4**(2), 135–148 (2013)
2. Becker, D., Munjere, A., Sawade, O., Massow, K., Thiele, F., Radusch, I.: Parking lot monitoring with cameras and lidar scanners. 2nd GI Expert Talk on Localization p. 17 (2016)
3. Bocquier, P.: World urbanization prospects: an alternative to the un model of projection compatible with the mobility transition theory. Demographic Research **12**, 197–236 (2005)
4. Camero, A., Toutouh, J., Stolfi, D.H., Alba, E.: Evolutionary deep learning for car park occupancy prediction in smart cities. In: International Conference on Learning and Intelligent Optimization. pp. 386–401. Springer (2018)
5. Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y.: Xgboost: extreme gradient boosting. R package version 0.4-2 pp. 1–4 (2015)
6. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Annals of statistics pp. 1189–1232 (2001)
7. Giuffrè, T., Siniscalchi, S.M., Tesoriere, G.: A novel architecture of parking management for smart cities. Procedia-Social and Behavioral Sciences **53**, 16–28 (2012)
8. Grassi, G., Jamieson, K., Bahl, P., Pau, G.: Parkmaster: An in-vehicle, edge-based video analytics service for detecting open parking spaces in urban environments. In: Proceedings of the Second ACM/IEEE Symposium on Edge Computing. pp. 1–14 (2017)
9. Ionita, A., Pomp, A., Cochez, M., Meisen, T., Decker, S.: Where to park? predicting free parking spots in unmonitored city areas. In: Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics. pp. 1–12 (2018)
10. Kim, K., Koshizuka, N.: Data-driven parking decisions: Proposal of parking availability prediction model. In: 2019 IEEE 16th International Conference on Smart

Cities: Improving Quality of Life Using ICT & IoT and AI (HONET-ICT). pp. 161–165. IEEE (2019)
11. Klandev, I., Tolevska, M., Trajanov, D.: Smart city: Public parking dashboard (2020)
12. Klappenecker, A., Lee, H., Welch, J.L.: Finding available parking spaces made easy. Ad Hoc Networks **12**, 243–249 (2014)
13. Lin, T., Rivano, H., Le Mouël, F.: A survey of smart parking solutions. IEEE Transactions on Intelligent Transportation Systems **18**(12), 3229–3253 (2017)
14. Litman, T.: Parking management. Victoria Transport Policy Institute (2016)
15. Mainetti, L., Palano, L., Patrono, L., Stefanizzi, M.L., Vergallo, R.: Integration of rfid and wsn technologies in a smart parking system. In: 2014 22nd International Conference on Software, Telecommunications and Computer Networks (SoftCOM). pp. 104–110. IEEE (2014)
16. Margreiter, M., Orfanou, F., Mayer, P.: Determination of the parking place availability using manual data collection enriched by crowdsourced in-vehicle data. Transportation research procedia **25**, 497–510 (2017)
17. Pullola, S., Atrey, P.K., El Saddik, A.: Towards an intelligent gps-based vehicle navigation system for finding street parking lots. In: 2007 IEEE International Conference on Signal Processing and Communications. pp. 1251–1254. IEEE (2007)
18. Shao, W., Zhang, Y., Guo, B., Qin, K., Chan, J., Salim, F.D.: Parking availability prediction with long short term memory model. In: International Conference on Green, Pervasive, and Cloud Computing. pp. 124–137. Springer (2018)
19. Shinde, S., Bhagwat, S., Pharate, S., Paymode, V.: Prediction of parking availability in car parks using sensors and iot: Sps. International Journal of Engineering Science and Computing **6** (2016)
20. Shoup, D.C.: Cruising for parking. Transport Policy **13**(6), 479–486 (2006)
21. Tayade, Y., Patil, M.M.: Advance prediction of parking slot availability with traffic and pollution updates for car parks in smart cities (2016)
22. Yang, C.F., Ju, Y.H., Hsieh, C.Y., Lin, C.Y., Tsai, M.H., Chang, H.L.: iparking–a real-time parking space monitoring and guiding system. Vehicular Communications **9**, 301–305 (2017)
23. Zhao, Z., Zhang, Y.: A comparative study of parking occupancy prediction methods considering parking type and parking scale. Journal of Advanced Transportation **2020**
24. Zheng, Y., Rajasegarar, S., Leckie, C.: Parking availability prediction for sensor-enabled car parks in smart cities. In: 2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP). pp. 1–6. IEEE (2015)