Journal of Internet Services and Applications

RESEARCH Open Access

Forecasting the carsharing service demand using uni and multivariable models

Victor Aquiles Alencar*† , Lucas Ribeiro Pessamilio†, Felipe Rooke†, Heder Soares Bernardino† and Alex Borges Vieira†

*Correspondence:
victoraquiles@ice.ufjf.br
†Victor Aquiles Alencar, Lucas Ribeiro Pessamilio, Felipe Rooke, Heder Soares Bernardino and Alex Borges Vieira contributed equally to this work.
Universidade Federal de Juiz de Fora, Rua Jose Lourenco Kelmer, 36036-900 Juiz de Fora, Brazil

Abstract
Carsharing is an alternative to urban mobility that has been widely adopted recently. This service presents three main business models: two of these models base their services on stations while the remainder, the free-floating service, is free of fixed stations. Despite the notable advantages of carsharing, this service is prone to several problems, such as fleet imbalance due to the variance of the daily demand in large urban centers. Forecasting the demand for the service is a key task to deal with this issue. In this sense, in this work, we analyze the use of well-known techniques to forecast a carsharing service demand. More in deep, we evaluate the use of the Long Short-Term Memory (LSTM) and Prophet techniques to predict the demand of three real carsharing services. Moreover, we also evaluate seven state-of-the-art forecasting models on a given free-floating carsharing service, highlighting the potentials of each technique. In addition to historical carsharing service data, we have also used climatic series to enhance the forecasting. Indeed, the results of our analysis have shown that the addition of meteorological data improved the models' performance. In this case, the mean absolute error of LSTM may fall by half, when using the climate data. When considering the free-floating carsharing service, and prediction for the short-term (i.e., 12 hours), the boosting algorithms (e.g. XGBoost, Catboost, and LightGBM) present superior performance, with less than 20% of mean absolute error when compared to the next best-ranked model (Prophet). On the other hand, Prophet performed better for predictions conducted on long-term periods.

Keywords: Car-sharing, Prediction, Deep-learning

1 Introduction

In the last few years, several studies have focused on the understanding of urban mobility and its applications [1–3]. The growing attention to urban mobility occurs due to its relationship with –practically– all of the daily tasks, as people labor, education, security, and health. These recent studies suggest that the increasing complexity within such tasks requires intelligent solutions for better resource management.

The current sharing economy phenomenon, which is an economic model mostly based on a peer-to-peer (P2P) mode, influenced the creation of several solutions for urban mobility. This occurs since people can easily share goods (e.g., vehicles), mainly through online platforms [4]. In this way, carsharing is an urban mobility solution that is receiving increasing attention from the academic community and industry [1, 2, 5, 6]. At a glance, the business models for carsharing are based on offering vehicles in a service area where users can book or use them at all times, without the inconveniences of owning a private vehicle [3]. In 2015, this kind of service handled more than 1.5 million users and 22 thousand vehicles in circulation in the Americas [3].

Carsharing services present three main business models. Two of these models use fixed stations while the remainder is free of fixed stations [6]. More in deep, we may call these models two-way, one-way, and free-floating models. In a two-way service model, also known as round-trip service, a vehicle can be booked at stations spread over a region. At the end of the rental, the vehicle must be returned to the initial station. The one-way model allows for the user to pick up a vehicle in a station and leave it at any station of the service. Finally, the free-floating model admits picking up a vehicle without the service station limitation and leaving this vehicle at any parking lot allowed by the service [5].

The user behavior of a carsharing system has already been addressed. We are aware of several works that describe the usage and demand patterns of these types of services [1, 2, 5]. In these recent works, researchers have shown results that the use of carsharing services follows a diurnal pattern and one of the biggest challenges related to the one-way and the free-floating carsharing models is the fleet imbalance over the service area. Fleet imbalance occurs when a large amount of vehicles is concentrated in areas/stations where the demand for service is low while other areas/stations with higher demand do not have enough vehicles to supply all users. This imbalance causes losses to the service operation, company profits, and users' dissatisfaction [7].

To deal with fleet imbalance, carsharing services providers must relocate their fleet, a problem known as Vehicle Relocation Problem (VReP). There are two main strategies to address VReP: user-based and operator-based. In the operator-based strategy, additional employees are used to relocating vehicles [8]. These relocations can be performed individually or by stork trucks and can be combined with vehicle maintenance and cleaning. In user-based strategies, the operator tries to influence the customer to start or finish their trip in different places than initially planned. For this, the operator uses different incentives or bonuses for customers. Trips to specific areas can be offered at lower rates or even at no cost when needed [9]. Despite the pros and cons of each strategy, both depend on accurate predictions for quick and effective fleet optimization.

In this sense, we analyze the use of well-known time-series forecasting models to predict the demand for real carsharing services. Most existing works in this area focus on a single carsharing service model [2, 10], and using a specific forecasting technique. In this work, on the other hand, we use seven time-series prediction techniques and their variants. Moreover, we use real data from the main three carsharing service models. More in deep, we first explore two forecasting models, the Long Short-Term Memory (LSTM) [10] and Prophet [11, 12], to predict the demand of three real carsharing services located in Vancouver city, Canada. Then, differently from our previous work [13], we further evaluate seven state-of-the-art forecasting models on a given free-floating carsharing service, highlighting their pros and cons. Moreover, we have also used here climatic series in addition to historical carsharing service data to enhance the forecasting, especially when the dataset presents missing data. Finally, we highlight that in this work, the performance of the forecasting models for short and long-term periods (i.e., 12 hours in the future and a week in the future) is evaluated.

In sum, our results show that the mean absolute error of the multivariate model of the LSTM —which uses both the carsharing historical data and the meteorological data— practically falls by half when compared to its univariate model. Moreover, the multivariate model of the LSTM is approximately 10% better, in terms of MAE, when compared to Prophet, for all of the three carsharing systems we evaluate.

The deeper studies we conduct on the free-floating carsharing system show that the boosting algorithms (e.g.XGBoost, Catboost, and LightGBM) have superior performance, with less than 20% of mean absolute error, when compared to the next best-ranked model (Prophet), for short-term predictions. Prophet, on the other hand, performs better when considering predictions for longer periods.

The remainder of this paper is organized as follows: in Section 2, we present and criticize the state of the art. In Section 3, we describe the data collection method and the data we use in this work. In Section 4, we briefly review each forecasting method and the evaluation method. In Section 5, we present the forecasting analysis for the carsharing demand of three real-world services. In Section 6, we deeper analyze the forecasting models for a free-floating carsharing system. Finally, Section 7 concludes the article.

2 Related work

As we previously stated, most of the existing previous works focus on characterizing and modeling the characteristics and usage patterns of carsharing services [1, 2, 5]. These works analyze Spatio-temporal service characteristics and clients' demands. For example, one can evaluate how the carsharing fleet demands differ between weekdays and weekends, indicating that forecasting models using inappropriate data can lead to bad results. To the best of our knowledge, our previous work [1] was the first to analyze and compare data from the three main carsharing service business models. Beyond the fleet demand, we have also evaluated the travel duration and the geospatial location demand for the service.

For both main strategies to address the VReP problem, accurate forecasting is a key step to a quick and effective fleet optimization. As expected, one may use several methods to address this task. For example, Uber, one of the most well-known transport systems of the new sharing economy age uses the LSTM (*Long Short-Term Memory*) [10] to forecast the demand for its services. LSTM has been chosen due to its scalability and prediction quality. The modified versions of LSTM in [10] improves its prediction accuracy, especially, while forecasting the demand during extreme events, such as holidays.

Prior work has already explored time series analysis in carsharing to forecast the fleet demand. For instance, Cocca et al. [2], investigates the demand for a free-floating service using state-of-the-art machine learning algorithms. The authors forecast the fleet demand using additional sources of data, as the past service demand, weather conditions, and socio-demographic data. According to their results, the Random Forest model presents a lower error rate while forecasting the fleet demand. In this case, the error varies from 10%, in case of using multiple sources of data, to 40%, in case of using only past fleet history. Differently from Cocca et al., we consider here three distinct carsharing service models and a wide number of machine learning and forecasting approaches.

Müller and Bogenberger [14] also investigated the time series analysis of a free-floating carsharing system, focusing on a short-term forecasting model. They have modeled the service time series (fleet demand) using ARIMA and investigated several time intervals. In this sense, they have evaluated the best interval to achieve good predictions. According to their results, the best performance to forecast the next week's demand is obtained using data from the previous three months.

In sum, forecasting data is a common topic for a wide number of research areas. Especially, time series has already been explored in industry and academia and, nowadays, there are several new methods. The methods we use in this work are well-known methods and serve as a substrate to the development of solutions for real-world situations. For example, Prophet has already been used in the literature and obtained a good performance when compared with the most used univariate time series methods for forecasting [11]. To forecast time series of temperature and precipitation, Prophet obtained competitive results among the classical models used on the area, especially when combined with a seasonal time series decomposition approach [12]. Other researchers used the Prophet model to forecast the Bitcoin value over time, where Prophet performed better than ARIMA [15].

Finally, we highlight that the three main carsharing service models are considered here and that we used real-world data of carsharing services from a large city in Canada. Moreover, we evaluate several forecast models: statistical-based, neural networks, and well-known time-series models.

3 Data collection methodology and dataset summary

In this work, we use real-world travel data collected from three different carsharing services in Vancouver, Canada. These carsharing services are Evo, Modo, and Car2Go, which follow, respectively, the one-way, the two-way, and the free-floating business model. The data crawling follows the methodology of our previous work [1], where we collected the fleet information and the car status of each carsharing service (i) using a publicly available API or (ii) by collecting data from a web page.

For instance, the Evo website [16] provides the location of the base stations and the information about vehicles, as its availability. The Modo API [17] returns the vehicles' information from each station, including their availability information and booking. As Car2Go is a free-floating service, its API [18] provides the start and end location of each vehicle (georeferenced), and the start and end times of each travel. In sum, for each service, we can infer when and where a user picks up a vehicle and leaves it. Then, in this work, we define travel (or trip) as a tuple containing the start/end locations, and the start/end timestamps of the travel.

In this sense, each service dataset contains the travel (trip) information which allows us to reconstruct the demand for the service. The dataset presents a minute timescale granularity, however, as a matter of clarity, we consider the service demand as the number of travels grouped by an hour, considering the travel starting period. For more details regarding the data collection, we point readers to [1].

For multivariate analysis, have obtained meteorological information using an API, called *World Weather Online* [19], to access the historical weather data. We collected temperature, wind speed, precipitation quantity, visibility, UV index, thermal sensation, and a classification of the weather for each day we analyze.

We have collected data from Evo and Modo from March 1, 2018, to July 15, 2018, a period of approximately five months. The period we have collected data from Car2Go starts on December 13, 2016, and ends on January 31, 2018, totaling one year and 49 days. Table 1 summarizes the data obtained from each service. The Car2Go service has the largest amount of data among those considered. It is also possible to note that even with the same collection period, Evo has a significantly higher number of trips compared to Modo.

The final datasets contain missing data on different periods due to problems in the data collection process (e.g., interruptions by lack of energy and processing). Thus, we properly selected a time interval for each carsharing service to avoid issues caused by these missing data during the training of the models. The period from June 23 to July 17 in 2018 has been used for EVO and Modo, and the entire year of 2017 has been considered for the Car2Go.

In Section 6, we further evaluate seven state-of-the-art forecasting models on the free-floating carsharing service, i.e., Car2Go, since it presents the largest dataset. Moreover, the Car2Go dataset presents the smallest number of missing data, ranging from December 13, 2016, to February 25, 2017.

Finally, all the datasets and source codes implemented in this work are public available[1].

4 Forecast methods

In this section, we briefly introduce the models and techniques used to forecast the carsharing system demand. Moreover, we present a short resume of each technique's implementation and how they were used. Finally, we show the metrics considered here to evaluate the performance of the models.

4.1 ARIMA and SARIMA

The Auto-Regressive Integrated Moving Average (ARIMA) and its derivations are widely used in the time-series forecast [2, 14, 15]. ARIMA is a univariate model that combines two models, namely an Auto-Regressive (AR) and the Moving Averages (MA) models, with an Integrated (I) part, which indicates that the values are not stationary. The AR model specifies a regression model where the dependent variable depends on his past values, and the AM model indicates the regression errors as a linear combination of past errors. SARIMA uses the same baseline of the ARIMA model, however, it includes a *Seasonality* part to improve forecasts with daily, weekly, or any other type of periodicity [2].

The ARIMA model has three parameters [2]: the number of time lags of the autoregressive model (p), the number of times the data must differentiate to obtain stationary data (d), and the number of moving average terms (q). As expected, SARIMA has the same parameters as ARIMA, plus a set of parameters to indicate the seasonality. Additionally,

Table 1 Quantitative summary of the dataset

Service	Trips	Vehicles
Evo	644 887	1 237
Modo	98 915	682
Car2Go	1 095 577	1 077

[1] http://modo.coop/api/, https://www.car2go.com/api/tou.htm last access: 14/05/2021

a parameter to indicate the number of periods in each season (m) is considered. We use the *pmdarima library*[2] in our computational experiments.

4.2 Prophet

Prophet is a statistical model developed by Facebook. This method has been designed to be an easy-to-use scalable method with a focus on time series that have strong seasonality patterns and suffer effects generated by holidays or extreme events [11]. Its forecasting model consists of three other models: (1) Trend, which is concerned with non-periodic changes in values; (2) Seasonality, which translates periodic changes in values, such as daily or weekly patterns; (3) Holidays, which represents the changes that may occur with the incidence of festive days or sudden changes in patterns. Thus, it can be considered an appropriate model for predicting human mobility behavior.

More specifically, Prophet is a decomposable time series model with three models, that can be defined by:

$$y(t) = g(t) + s(t) + h(t) + e_t \tag{1}$$

where $g(t)$ represents the trend model, that is a modified linear or logistic model to describe the growth of the time-series values; $s(t)$ is the seasonality model, which is a Fourier series to provide a flexible model to predict periodic changes; $h(t)$ models the holidays, where each hour of a holiday is mapped to influence in the forecast; and the error parameter e_t represents any changes not included in the model. The trend model is defined by the user. More information can be found in [11].

4.3 Gradient boost

Gradient Boosting is an ensemble method that produces an improved model by fitting a new predictor with the residual error made by the previous predictor [20]. The Gradient Boosting Models contains three main components: a loss function to be optimized, a weak learner to perform predictions, and an additional policy to include the models aims to minimize the loss function. The loss function used depends on the type of problem being solved and the squared error function is an example. Usually, the weak learners in gradient boosting are decision trees.

In this work, we use three gradient boosting models: XGBoost, CatBoost, and LightGBM. XGBoost is scalable and fast, using techniques to data compress and cache access patterns. Also, it deals with sparse data and uses approximate tree learning [21]. CatBoost introduces the ordered boosting approach that is an alternative to the classical boosting algorithms and processes categorical features [22]. LightGBM is composed of two techniques: the Gradient Based One-Sided Sampling, which excludes small proportions of data that have small gradients; and the Exclusive Feature Bundling, which bundles mutually exclusive features to reduce its number [23].

4.4 LSTM

The *Long Short-Term Memory* (LSTM) is a type of neural network that stands out for its time series prediction [2, 10]. It is based on a Recurrent Neural Network (RNN) which maintains a short-term memory to influence the forecast of the next time interval. Thus, during the training, it determines the time interval used as memory and the prediction

[2] https://alkaline-ml.com/pmdarima/index.html, last access: 08/26/2020

horizon. The data model is organized according to the following shape: [batch size, time stamps, columns]. The batch size indicates the number of instances. The number of time steps represents how many timestamps each batch contains for short-term memory and, finally, the number of columns represents how many columns of data are assigned for each timestamp.

4.5 Performance metrics

Finally, for performance comparison purposes, all results were evaluated using the metrics RMSE (*Root Mean Square Error*) and MAE (*Mean Absolute Error*). These are metrics widely used in the time-series literature.

RMSE can be defined as:

$$\mathrm{RMSE}(\mathbf{Y}, \hat{\mathbf{Y}}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2} \tag{2}$$

where n is the number of samples, Y_i is the i-th expected value and $\hat{Y}_i$ is the i-th predicted value. In other words, RMSE is the Euclidean distance between the expected and predicted values scaled by a factor of $1/\sqrt{n}$.

Similar to RMSE, MAE measures the mean of the difference between the expected and predicted values, but 1-norm is used in this case. The MAE metric is defined as:

$$\mathrm{MAE}(\mathbf{Y}, \hat{\mathbf{Y}}) = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i| \tag{3}$$

5 Forecasting the carsharing demand for three real-world services

In this section, we evaluate the use of the Long Short-Term Memory (LSTM) and Prophet techniques to predict the demand for three real carsharing services. For the univariate model (i.e., Prophet and LSTM), we use only the historical demand of the carsharing services to train the models. For the multivariate model (i.e., LSTM), we also the meteorological historical data of the city.

For Prophet, which is a univariate model, we have split the dataset into two subsets: a training set, with 80% of the data, and a testing set, with the 20% remaining data. In the following analysis, we present the Prophet predicted values using both subsets, i.e., the prediction on the training set and the prediction on the test set. LSTM may be used as a univariate or a multivariate model. For the LSTM, we have split the dataset into three subsets: a training set, with 60% of data, a validation set with 20%, and a testing set with the remaining 20% of data. We used an additional subset for LSTM due to the parameter tuning through the *Grid Search* method. For both models, we evaluate the MAE and RMSE using the test sets.

Here, we use Prophet to forecast the demand for a given period considering all the input data of the training set or the test sets. For the LSTM, on the other hand, we consider data from the past 24 hours to forecast the demand for the next 12 hours.

In the following analysis, we set up the default Prophet parameters. We have also included the Canada Day holiday, that occurred on July 1, 2017, and July 2, 2018. The LSTM parameters have been chosen by a Grid Search process and can be seen in Table 2.

The remainder of this section presents the forecasts and the performance evaluation of the time-series considered here when applying Prophet (Section 5.1) and LSTM

Table 2 LSTM parameters summary

Model	Parameter List
Single Variable	{ "layers" = [LSTM_layer(50 Nodes, dropout=0.2), LSTM_layer(50 Nodes, activation=relu), Dense_layer(12 Nodes)] "optimizer": RMSprop(clipvalue=1.0), "loss": "mae", "epochs": 15, "evaluation_interval":50, "validation_steps": 50, "batch_size": 256 for the three Real-World services and 64 for the Free-floating comparison,}
Multi Variable	{ "layers" = [LSTM_layer(80 Nodes, dropout=0.2), LSTM_layer(80 Nodes, activation=relu), Dense_layer(12 Nodes)] "optimizer": RMSprop(clipvalue=1.0), "loss": "mae", "epochs": 15, "evaluation_interval":50, "validation_steps": 50, "batch_size": 256 for the three Real-World services and 64 for the Free-floating comparison,}

(Section 5.2). In the latter, univariate and multivariate variants of the problem are solved. Finally, the results obtained are discussed in Section 5.3.

5.1 Prophet

Figures 1, 2 and 3 present the carsharing services demand and the predicted values, when using Prophet. The y-axis presents the carsharing system demand (i.e., the total number of travels) and the x-axis indicates the instants of time (with one-hour granularity). In this figure, the black points represent the real carsharing demand observed in our dataset. The curve in blue, labeled as "Prediction on training set", represents the Prophet prediction when considering the training set. The dashed orange line, labeled as "Prediction on test set", refers to the prediction when considering the 12 hours of the test set. In addition, the range of uncertainty in the forecasts is displayed along with the blue/orange curves.

Figure 1 presents the carsharing forecast demand for the Evo carsharing system when using the Prophet model. In general, one can notice that the forecasting follows the real data. Moreover, the uncertainty Prophet is not negligible. Finally, the missing data (i.e., the days that the log file does not have complete data) changed the initial behavior of the model. However, when the time series is complete, the model fits the existing data correctly. The forecast on the training set also followed the pattern of the actual data, with a local maximum value for the morning period, followed by a decrease in demand during the afternoon to result in a global maximum in the future. This model reflects the demand behavior of the service as it is a one-way service and, consequently, its trips are normally short. There are not many vehicles occupied at the same time along the day, except during the peak periods.

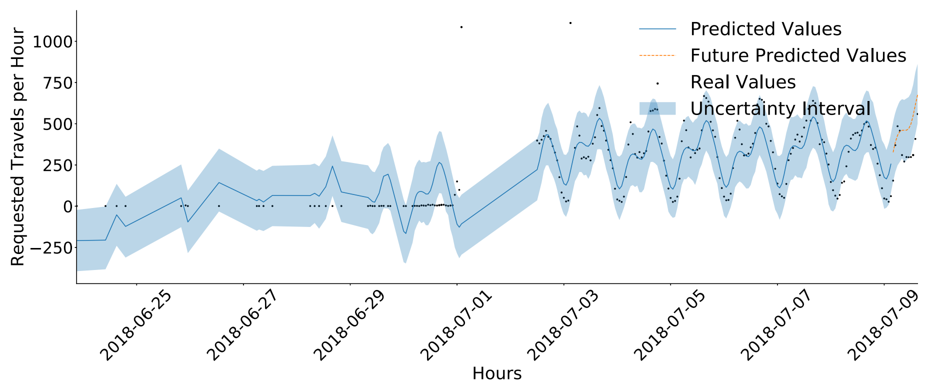

Fig. 1 12-hours forecast using the Prophet model for the Evo

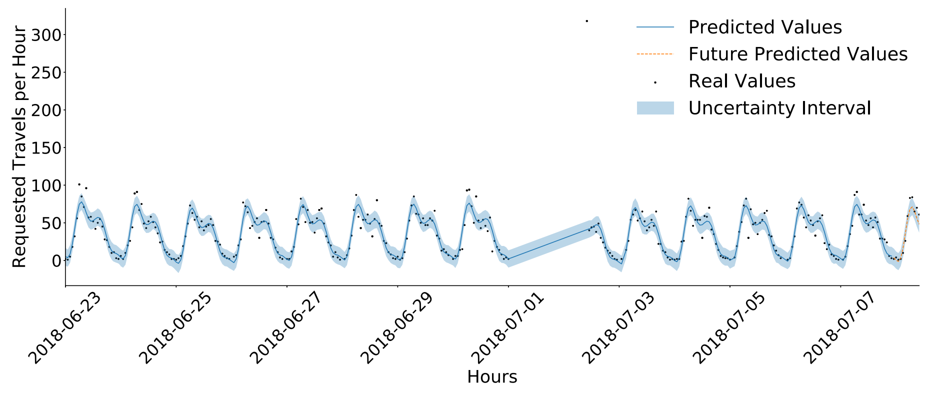

Fig. 2 12-hours forecast by Prophet model for Modo

Figure 2 shows the forecasting performance of the Prophet for the Modo carsharing system. Initially, one can observe less missing data in this time series when compared to Evo (Fig. 1). We can see the daily pattern that the model adopted has an overall maximum in the morning period followed by a decrease, which results in a local maximum at noon that continues decreasing throughout the day. The forecast on the testing set (12 hours), on the rightmost part of the figure, highlights the morning peak, with the initial decay of the afternoon. In Modo, users tend to pick up the vehicles in the morning and use them throughout the day. This justifies the peak of demand in the mornings and lower demand at noon. It is important to note that Fig. 2 shows a point with a high demand value. This point has not been removed from our analysis as it is the morning after the Canada Day holiday and its inclusion improved the model's performance.

Figure 3 shows the results obtained by Prophet when forecasting the carsharing system demand for the Car2Go carsharing system. The Car2Go dataset has less missing data when compared to the dataset of the other two services. First, one can see that the model's behavior is similar to Evo (Fig. 1) regarding the incidence of local maximums. However, the increase of the usage along the day reaches a local peak around noon and then increases its demand throughout the day. As result, the forecast on the test set reflected the constant growth in demand that starts in the morning and reaches a global

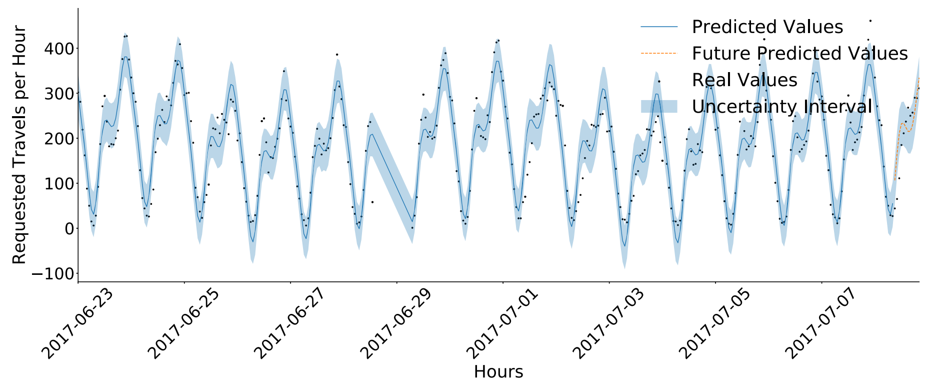

Fig. 3 12-hours forecast by Prophet model for Car2Go

maximum at the noon. As Car2Go is a free-floating service, it is used for short trips and, consequently, its demands are similar to Evo.

5.2 LSTM

In this section, we present the results we obtain when using the univariate and multivariate LSTM models. Figures 4, 5 and 6 show the real dataset values and predicted curves for the testing data (12h). For each carsharing service, one can observe that both univariate and multivariate models present similar qualitative patterns.

In both cases, the univariate and the multivariate LSTM models follow the actual service demand. However, the univariate model (shown in Fig. 4) presents a higher error. Especially, at the beginning of the time interval, where the carsharing system presents a growing demand (from 0 to 8), the univariate model did not correctly forecast the expected values and the peak of demand. The curve of the multivariate model fits the data better than the univariate one, especially for the first peak/valley of demand.

Figure 5 shows the results of LSTM for the Modo carsharing system. Similar to the previous case, the curves of both models are similar to the actual data. The univariate model, once more, presents worse performance. In this case, the univariate model did not correctly predict the ending of the actual time series, resulting in a higher error value. The multivariate model can predict well all the actual data, reducing the error in general.

We also evaluate the performance of LSTM using the Car2Go data. As shown in Fig. 6, the univariate and the multivariate LSTM models present similar performance for this carsharing service, and, visually, both models present good results.

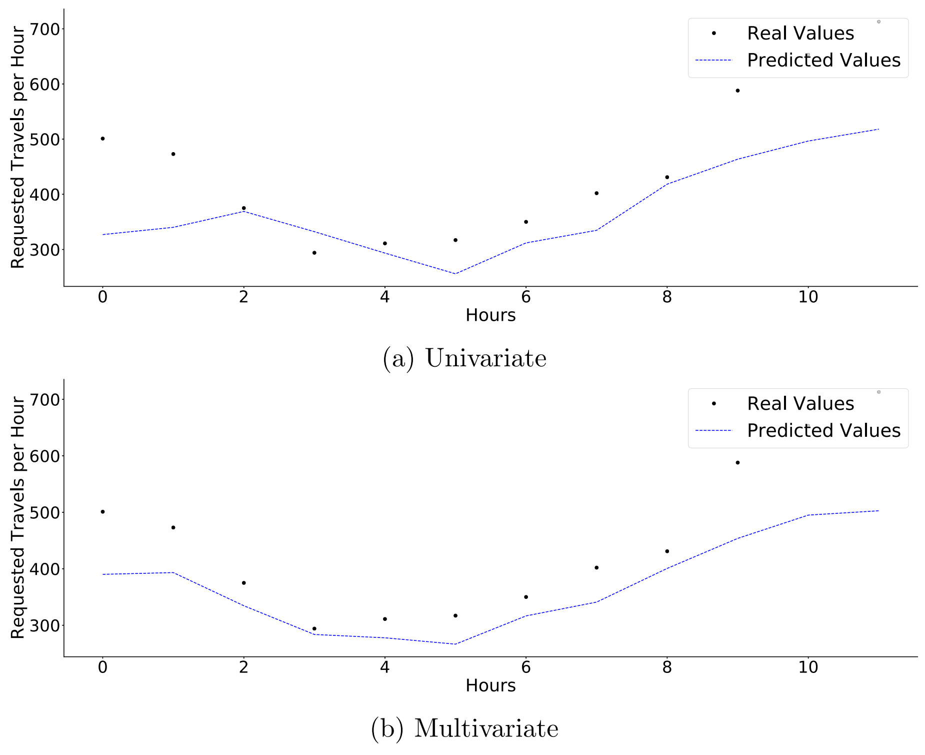

Fig. 4 12-hours forecast using the LSTM model for the Evo

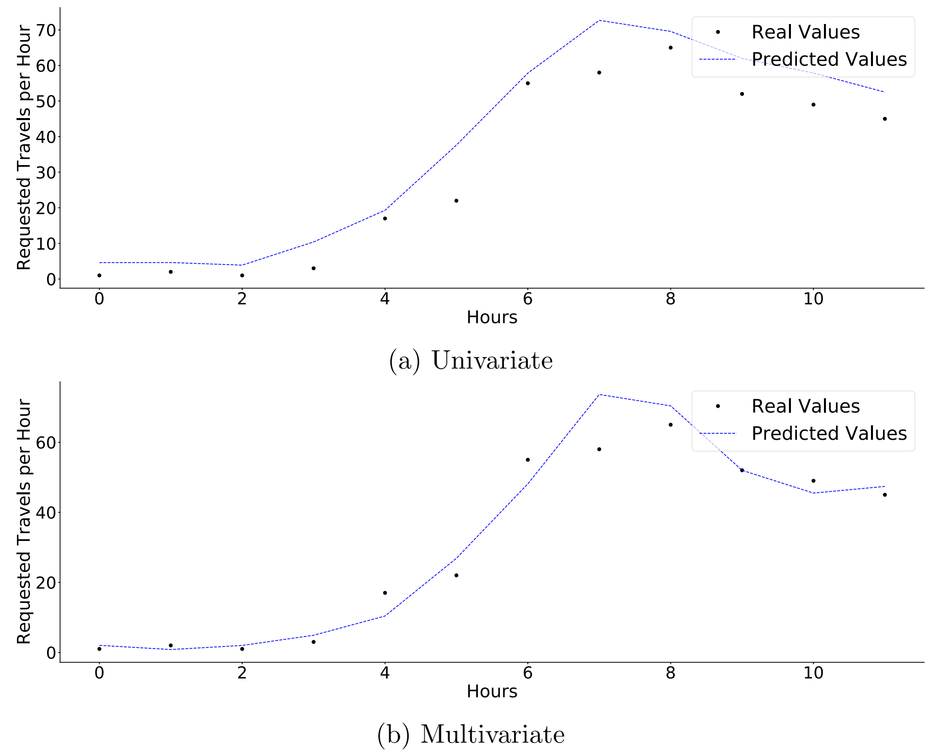

Fig. 5 12-hours forecast using the LSTM model for the Modo

5.3 Summary and discussion

Table 3 shown RMSE and MAE for the three services evaluated here, and the mean and standard deviation values of the predicted demands. The best performance, for each metric, is shown in bold font. The Prophet results, shown in the Table 3, evidence that the model performed well for the RMSE. In this case, low RMSE values represent how well the forecast fits the curve shape. Propet also shows a good performance for Evo. Despite the missing data in the time series of this service, it has presented the performance among the other services. The mean absolute error (MAE) of Modo and Car2Go is low. This may indicate the reliability of the model to these services. Evo presents a higher MAE, possibly due to the missing data.

The univariate LSTM model presents good performance, considering the RMSE, for predicting the Car2Go and Modo demands. This confirms the visual tests of the previous figures. The RMSE for Evo, in this case, is practically three times worse than the other services. The MAE corroborates these observations: the forecasting to Modo and Car2Go using LSTM have low absolute errors, which increases their reliability.

The MAE values the multivariate LSTM model presents are very close to the values the univariate model presents, as shown in Table 3. The only exception occurs for the Car2Go service that has practically half the MAE value for the multivariate model. Despite de similar behavior, the extra data to the multivariate model enhances the performance of the LSTM. In this case, the forecasting for Evo and Modo has been benefited.

When we compare the univariate and the multivariate LSTM models, the multivariate model shows better performance. For both cases, visually and quantitatively, the use of additional sources by the multivariate model —as the meteorological data— enhances

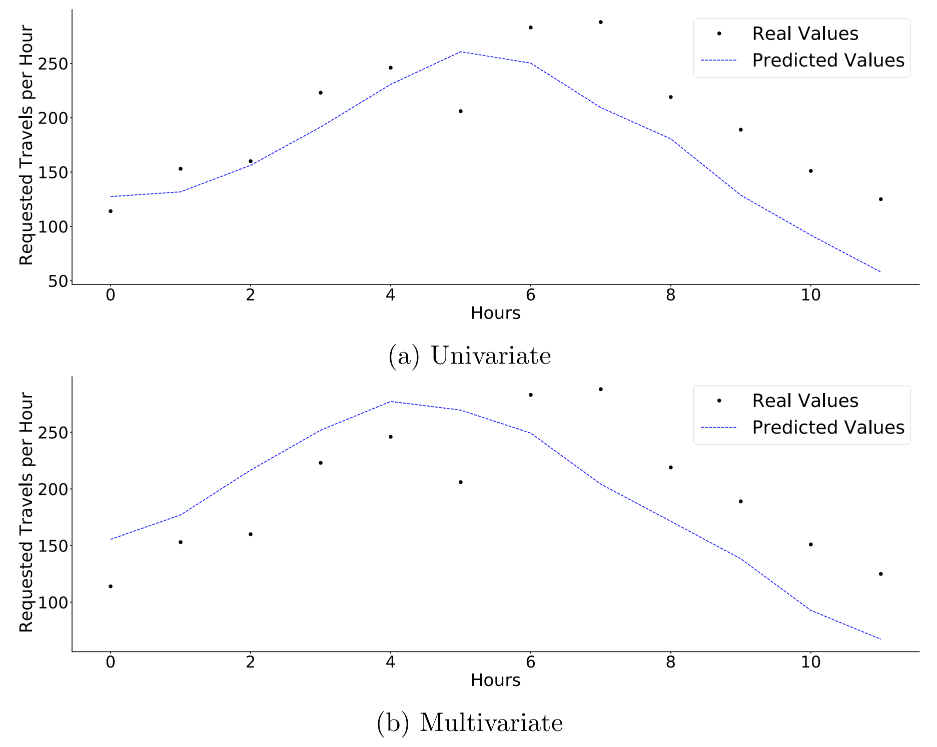

Fig. 6 12-hours forecast using the LSTM model for Car2Go

the forecasting quality. This is especially notable for Car2Go, which presents the dataset with a major lack of data between the three carsharing services. We also note a big difference in performance when we compare Prophet and LSTM. In general, the LSTM models outperform Prophet. The only case Prophet presents a better result refers to the Modo carsharing system.

6 Forecasting the demand for free-floating carsharing model

In this section, we further evaluate seven state-of-the-art forecasting models on the free-floating carsharing service, i.e., Car2Go. As we previously commented, the Car2Go has

Table 3 Values obtained by the models analyzed here

		Uni LSTM	Multi LSTM	Prophet
Metric	**Dataset**			
MAE	Evo	183.338071	**170.226942**	193.025442
	Modo	17.152931	16.968548	**6.987489**
	Car2Go	61.127329	**32.718110**	35.915285
RMSE	Evo	249.017479	243.963300	**215.351670**
	Modo	27.809029	27.505195	**9.255800**
	Car2Go	74.831262	**45.254456**	46.163827
MEAN	Evo	434.421569	434.421569	539.368923
	Modo	40.275194	40.275194	33.113844
	Car2Go	166.947115	166.947115	176.974592
STD	Evo	151.366257	151.366257	119.086183
	Modo	27.649906	27.649906	23.471948
	Car2Go	93.852766	93.852766	101.982915

The best MAE and RMSE values are highlighted in boldface the largest dataset and presents the smallest number of missing data. Moreover, we evaluate the performance of the forecasting models for short and long-term periods (i.e., 12 hours in the future and a week in the future), highlighting the pros and cons of each technique.

The seven models we consider in this section can be grouped as: (i) the statistical models, namely, ARIMA, SARIMA, and Prophet; and (ii) the multivariate models, namely Gradient Boost and LSTM. We split Car2Go data into two subsets, namely, training and test sets, with 80% and 20% of the original dataset, respectively. We use the test to conduct the analysis.

The first analysis consists of forecasting the carsharing system demand for the next 12 hours of a given moment. This test shows how well models can predict the short-term periods. We have chosen 12 hours period as it represents a daily pattern usage and services managers can use this information to better plan their services, re-allocating the fleet in strategic locals.

After each hour of the service, the prediction window slides. We add the predicted value for the current hour to the model input. Then, we predict the next period of interest. We call this process as *auto-feed process*.

For the statistic models, the auto-feed process has been performed by retraining the model with the new predicted data to predict the next hour. The other models, who accept multiple features, used the last 24 hours as features to predict the carsharing system demand and, again, they were auto-feed during each prediction.

The second analysis consists of predicting the next week of a given moment. This test evaluates the quality of the models for long-term periods. We have chosen a week period due to the seasonality of the demand (i.e., the same weekday tends to have similar demands, regardless of the week of the month). The forecasting routine is similar to the previously present, but the model predicts the next week after being trained with the previous data.

All the parameters have been chosen using a Grid Search process. The parameters of ARIMA have been set up as (p, q, d) = (2, 0, 3). The parameters of SARIMA have been set up as (p, q, d) = (2, 0, 3); (P, D, Q) = (2, 1, 1), and m = 12. Also, we have added the holidays that occurred during the data collection in Prophet, as it considers festive events as a parameter. Within the selected interval there are four holidays: Christmas day, on December 25, 2016; New Year, on January 1, 2017; Family Day, on February 13, 2017; and Valentine's day on February 14, 2017.

Table 4 presents the remaining of Prophet parameters. For further details on Prophet can be found in its official documentation[3].

Table 5 presents the parameters used here for Gradient Boost. More details about the libraries we use for the experiments with Gradient Boost methods can be found in their respective documentations[4,5,6].

When using LSTM, the data was split into three subsets as follows: 60% for training, 20% for validation, and 20% for testing. All LSTM models were trained using 30 epochs and 50 steps for training. Also, 50 steps were used for validation. For the prediction of the next 12 hours (i.e., short-term forecasting), the LSTM models use two layers with 220 nodes, the

[3]https://facebook.github.io/prophet/docs/quick_start.html, last access: 10/08/2020
[4]https://xgboost.readthedocs.io/en/latest/parameter.html, last access: 10/08/2020
[5]https://catboost.ai/docs/concepts/python-reference_parameters-list.html, last access: 10/08/2020
[6]https://lightgbm.readthedocs.io/en/latest/Parameters.html, last access: 10/08/2020

Table 4 Prophet parameters

Parameter	Value
changepoint_range	0.9
n_changepoints	25
changepoint_prior_scale	0.5
holidays_prior_scale	30
daily_seasonality	true
weekly_seasonality	true
yearly_seasonality	false
growth	linear
seasonality_mode	multiplicative

second layer uses the ReLU activation function and l2 regularization with weight 0.01 as a Recurrent and Kernel Regularizer. After these layers, the model has a final dense layer with 12 nodes. For the model that forecasts the next week (i.e., long-term forecasting), the model uses two layers with 160 nodes, the second layer uses the ReLU activation function and l2 regularization with weight 0.01 as a Recurrent and Kernel Regularizer. After these layers, the model has a final dense layer with 1 node. As previously discussed, we used the predicted values as input when forecasting the next week (168 Hours). As the model depends on the determination of a short-term memory period and a forecast horizon, we defined the last 24 hours to determine the forecast for the next 12 hours. The optimizer used was RMSprop with a clipvalue equals to 1.0.

For all the methods considered here, the non-commented parameters assumed their default value. The remainder of the section presents the results obtained for both computational experiments, namely, the prediction of 12 hours demand (Section 6.1) and one-week demand (Section 6.2). The test set is composed of data from the same time interval: the period between 03:00 of February 11, 2017, and 17:00 of February 25, 2017.

6.1 Short-term forecasting

Figures 7 and 8 present a visual example of the short-term carsharing demand forecasting, i.e., the number of requested travels in a given hour of day. In Fig. 7, we present the ARIMA, SARIMA, and Prophet forecasting results, while in while Fig. 8, we show the results CatBoost, LightGBM, XGBoost, and LSTM achieve. Both figures display 12 hours. Moreover, the dots represent the real data values, and the lines show the values models predict.

First, as shown in Fig. 7, ARIMA and SARIMA models do not perform well. Indeed, the predicted data for both models do not visually fit the actual data. On the other hand, the Prophet model presents a better performance when predicting the initial hours of the time series, as shown in this figure.

Table 5 Gradient Boost parameters

Model	Parameter List
XGBoost	{ 'colsample_bytree': 1, 'eval_metric': 'rmse', 'learning_rate': 0.005, 'max_depth': 10, 'min_child_weight': 8, 'n_estimators': 2000, 'objective': 'reg:squarederror', 'subsample': 0.5 }
CatBoost	{ 'iterations': 15000, 'learning_rate': 0.001, 'max_depth': 3, 'num_leaves': 31, 'loss_function': 'MAE', 'eval_metric': 'MAE'}
LightGBM	{ 'bagging_fraction': 0.7, 'bagging_freq': 10, 'feature_fraction': 0.9, 'learning_rate': 0.001, 'max_depth': 8, 'n_estimators': 40, 'num_iterations': 15000, 'num_leaves': 32 }

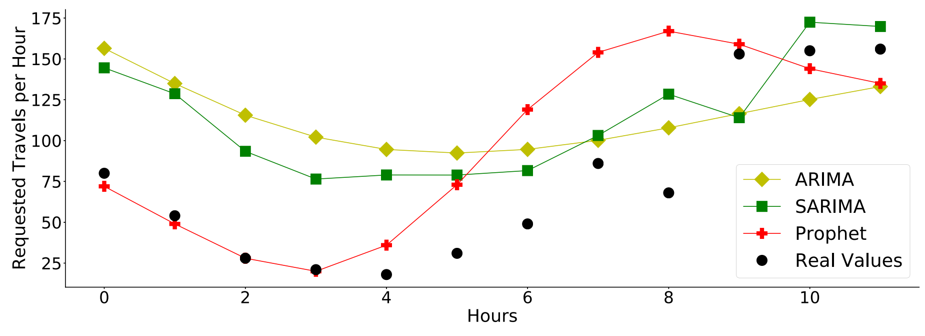

Fig. 7 Statistical models carsharing demand prediction for short-term (12-hours)

All Gradient Boosting models performed similarly, as shown in Fig. 8. These models fit well the shape of the real data, with fewer errors for the first 6 hours. The LSTM also fits well the shape of the real data. However, when compared to the Gradient Boosting models, LSTM presents higher errors during the initial period of prediction.

Figure 9 shows the MAE values for all the models and test sets we have evaluated. The performance of ARIMA has not been included in this figure due to visualization issues. Moreover, ARIMA presents the worst performance among all the models we analyze, whit a MAE of 94.451198. According to this figure, CatBoost, LightGBM, and XGBoost performed similarly and, in this case, they present the best performance among all the models tested here. However, the error of these models slightly increases over time. On the other hand, Prophet and SARIMA errors are practically constant, which is an important feature for forecasting. Finally, LSTM achieved the second-worst performance among the techniques tested here and the worst results in this figure, with an increasing error over time.

Table 6 shown the error metrics of each model, and the mean and standard deviation of predicted demand values. The best performances (lower error) for each metric are in boldface. The Gradient Boosting models show a constant pattern of values between them, with small values for every metric in comparison with the other models. ARIMA and LSTM found the worst results. Prophet has the best performance among the statistical models, followed by SARIMA.

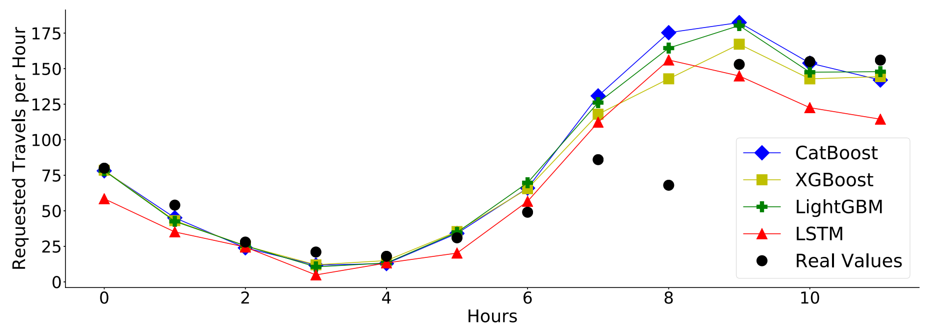

Fig. 8 Machine Learning carsharing demand models prediction for short-term (12-hours)

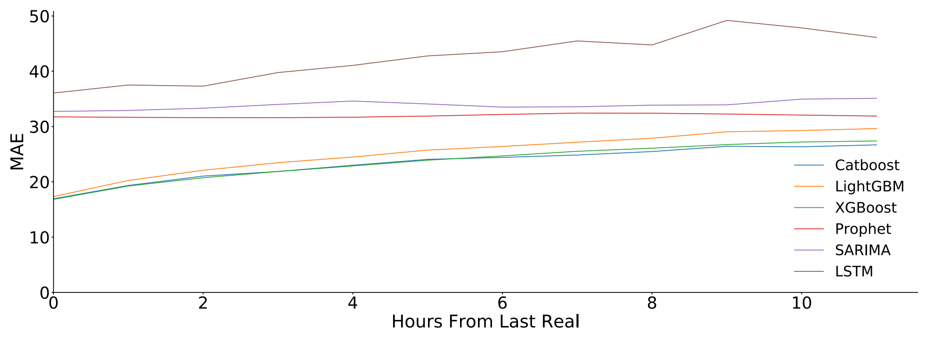

Fig. 9 Model errors for the short-term (12-hours) prediction

6.2 Long-term forecasting

As in Section 6.1, where we presented the short-term period forecasting of the carsharing system demand, Figs. 10, 11, 12, 13, 14, 15 and 16 show the long-term carsharing demand forecasting results. Again, the dots represent the real values of the dataset and the lines are the values each model predicts. As shown in this figure, ARIMA poorly performs to forecast the long-term period. On the other hand, SARIMA performs well, especially for the first half of the period considered here. Prophet (Fig. 12) performs well for the entire period. Indeed, this model fits well the curve for almost all the long-term periods.

Different from the short-term periods forecasting, the Gradient Boosting models degenerated quickly. As shown in Figs. 13, 14 and 15, the Gradient Boosting methods do not performed well after –about– 75 hours. According to Fig. 16, LSTM performed similarly to Prophet. However, LSTM does not found good predictions for the initial 24 hours.

Figure 17 shows the MAE values for the models we evaluate in this work. Again, as occurred in the short-term forecasting case, we have not included the MAE values for ARIMA in this figure for better visualization. Similar to the previous situation, the performance of ARIMA is worst than those achieved by the other models. Prophet obtained the best results for predictions after 40 hours. Especially, the MAE of the Prophet predictions is almost constant, making this model the most reliable among those evaluated here. LSTM obtained the worst performance at the beginning of the prediction time interval, but its relative performance becomes better when the prediction window is larger. The Gradient Boosting models performed well at the beginning, but their errors increase over time. In general, the Gradient Boosting techniques achieved one of the worst overall performances. Prophet obtained the best demand predictions for the long-term when compared to the methods tested here.

Table 6 Summary for the carsharing demand prediction in the short-term (12 hours)

	MAE	RMSE	MEAN	STD
Catboost	**23.371369**	**32.704312**	155.964375	92.419203
XGBoost	23.591305	33.270057	156.686951	92.391609
LightGBM	25.231280	35.210025	156.390899	92.124506
Prophet	31.956735	41.979346	147.266962	85.300089
SARIMA	33.884065	43.560814	114.444472	18.985461
LSTM	42.615119	58.792255	125.105176	73.699453
ARIMA	94.451198	114.812688	114.444472	18.985461

The best MAE and RMSE values are highlighted in boldface

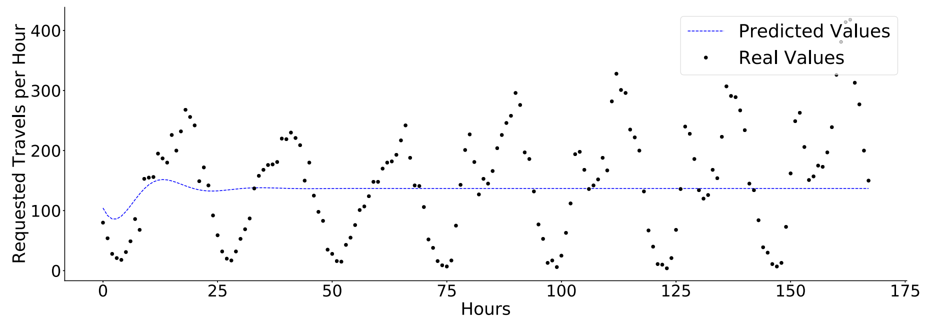

Fig. 10 ARIMA carsharing demand prediction for long-term (one week)

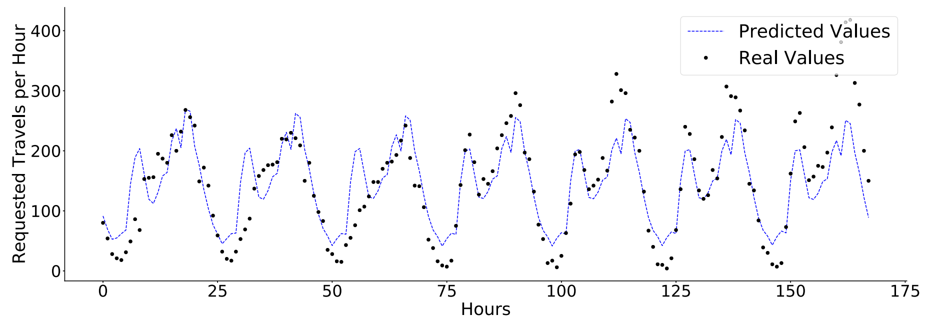

Fig. 11 SARIMA carsharing demand prediction for long-term (one week)

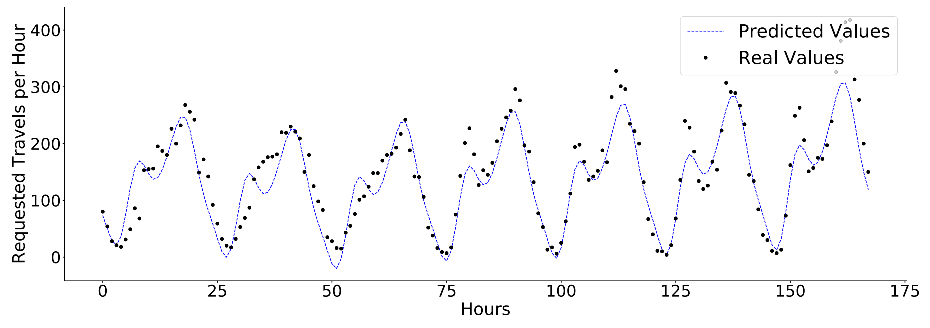

Fig. 12 Prophet carsharing demand prediction for long-term (one week)

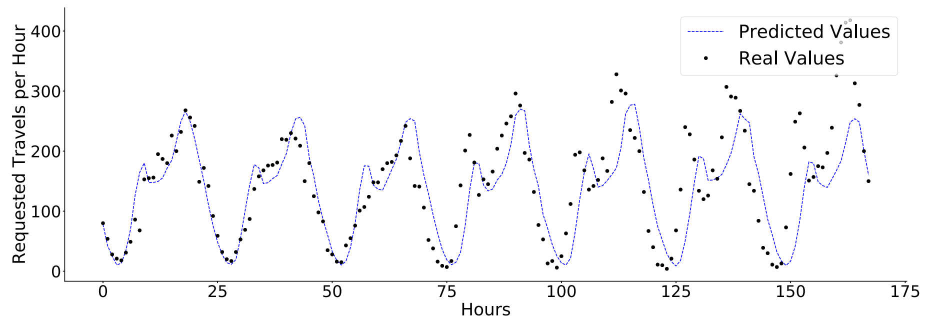

Fig. 13 LightGBM carsharing demand prediction for long-term (one week)

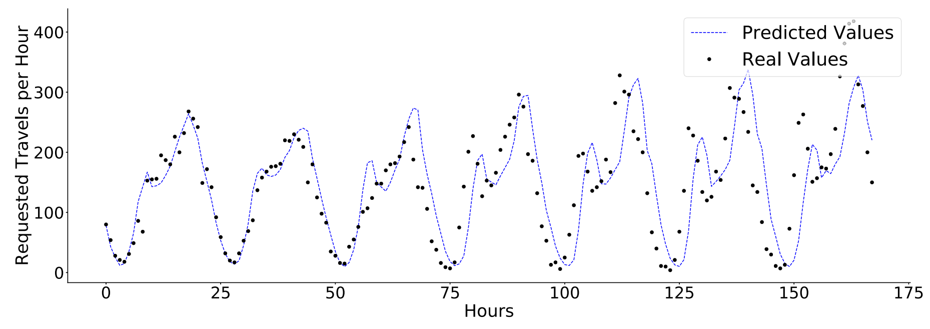

Fig. 14 XGBoost carsharing demand prediction for long-term (one week)

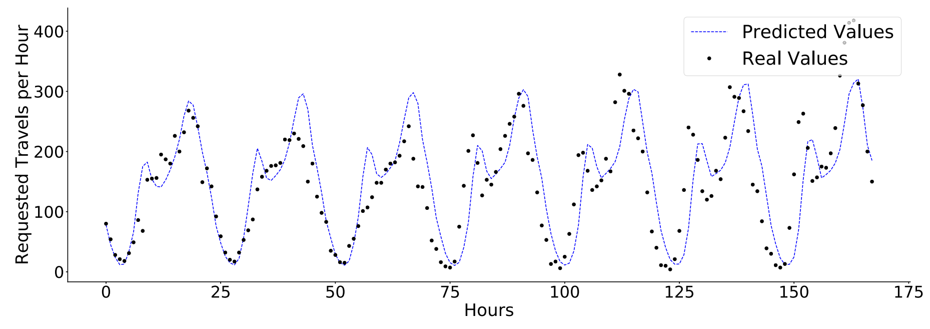

Fig. 15 CatBoost carsharing demand prediction for long-term (one week)

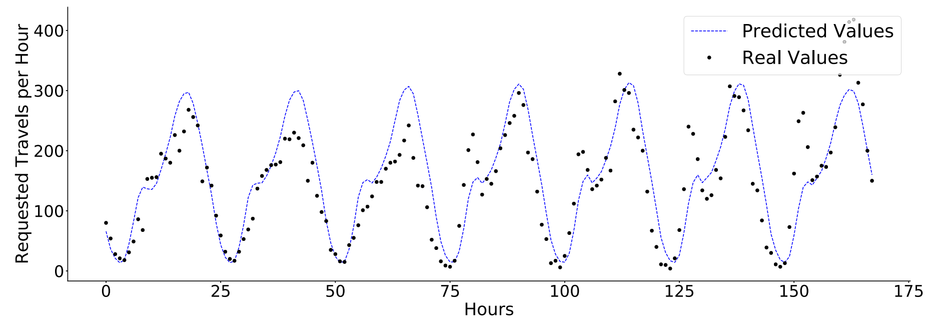

Fig. 16 LSTM carsharing demand prediction for long-term (one week)

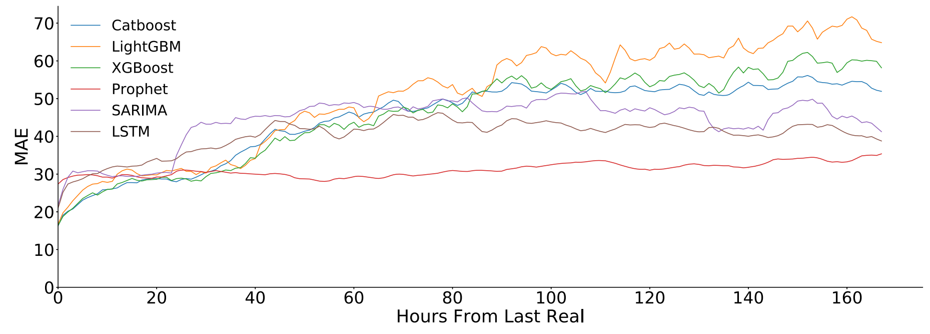

Fig. 17 Model errors for the long-term (one week) prediction

Table 7 Summary for the carsharing demand prediction in the long-term (one week)

	MAE	RMSE	MEAN	STD
Prophet	**31.158301**	**40.309776**	164.575212	94.899747
LSTM	40.232423	52.256360	156.020799	90.698742
SARIMA	44.108337	59.216509	136.845372	13.360609
Catboost	44.906164	59.831549	154.427596	89.016378
XGBoost	45.822976	61.749803	148.905914	86.049995
LightGBM	50.879445	69.052369	143.345790	84.880683
ARIMA	86.378079	105.126036	136.845372	13.360609

The best MAE and RMSE values are highlighted in boldface

Table 7 presents the MAE and RMSE of the predictions, and the mean and standard deviation values (for the forecasted demand). The best performances (lower errors) for each metric are in boldface. Prophet obtained the best overall performance. The Gradient Boosting techniques and Arima found the worst results. Sarima performed similarly to Catboost. LSTM obtained the second-best performance for the long-term predictions (one week), differently from the results of the short-term case (12h).

7 Conclusion

In this article, we have analyzed the use of well-known predictive models to forecast the demand for real carsharing services. First, we evaluate two models – the Long Short-Term Memory (LSTM) and the Prophet– for three distinct carsharing services models. Then, we evaluate seven state-of-the-art forecasting models on a given free-floating carsharing service.

In sum, when comparing LSTM and Prophet for all three carsharing services, the multivariate LSTM which also considers the use of meteorological data presents better performance. In this case, the mean absolute error of the multivariate LSTM is practically half the value of the univariate LSTM model. Moreover, the multivariate LSTM is 10% better than Prophet, considering the mean absolute error of predictions. For the specific free-floating carsharing service analysis, the Gradient Boosting methods present better performance for predicting short-term periods (i.e., 12 hours). On the other hand, the Prophet achieves the best performance when forecasting long-term periods (i.e., one week). In the case of long-term periods, in this work, simple methods as SARIMA have performed better than the Gradient Boosting models.

In future work, we intend to include geolocation characteristics to better determine the demand of the service, grouped by service operation. Moreover, we also intend to use optimization methods, jointly the forecast models, to support the decision-making procedures. Finally, we also intend to apply this research in similar contexts, as for bike-sharing services. We intend to study the characteristics of this service and how the forecasting methods we apply to carsharing are effective or not to this service.

Abbreviations

MAE: Mean Absolute Error; RMSE: Root Mean Squared Error; STD: Standard Deviation; LSTM: Long Short-Term Memory; XGBoost: Extreme Gradient Boosting; CatBoost: Category Boosting; LightGBM: Light Gradient Boosted Machine; ARIMA: Auto-Regressive Integrated Moving Average; SARIMA: Seasonality Auto-Regressive Integrated Moving Average; VRep: Vehicle Relocation Problem

Acknowledgements

Authors thanks the support of Capes, CNPq and Fapemig.

Authors' contributions

All authors read and approved the final manuscript.

Funding
This research is partially funded by CNPq, CAPES, FAPEMIG, and CGI/FAPESP.

Availability of data and materials
Please contact author for data requests.

Declarations

Ethics approval and consent to participate
Not applicable

Consent for publication
Not applicable

Competing interests
The authors declare that they have no competing interests.

Received: 26 October 2020 Accepted: 20 July 2021
Published online: 04 August 2021

References
1. Alencar VA, Rooke F, Cocca M, Vassio L, Almeida J, Vieira AB. Characterizing Client Usage Patterns and Service Demand for Car-Sharing Systems. Inf Syst. 2021:98.
2. Cocca M, Teixeira D, Vassio L, Mellia M, Almeida JM, Couto da Silva AP. On Car-Sharing Usage Prediction with Open Socio-Demographic Data. Electronics. 2020;9(1):72.
3. Shaheen SA. Mobility and the Sharing Economy. Transport Policy. 2016;51(Supplement C):141–2. https://doi.org/10.1016/j.tranpol.2016.01.008.
4. Hamari J, Sjöklint M, Ukkonen A. The Sharing Economy: Why People Participate in Collaborative Consumption. J Assoc Inf Sci Technol. 2016;67(9):2047–59.
5. Boldrini C, Bruno R, Conti M. Characterising Demand and Usage Patterns in a Large Station-Based Car Sharing System. In: IEEE Conference on Computer Communications Workshops (INFOCOM Workshops). IEEE; 2016. p. 572–7. https://doi.org/10.1109/infcomw.2016.7562141.
6. Nourinejad M. Dynamic Optimization Models for Ridesharing and Carsharing. Master's thesis, University of Toronto. 2014.
7. Illgen S, Höck M. Literature Review of the Vehicle Relocation Problem in One-Way Car Sharing Networks. Transp Res B Methodol. 2019;120:193–204.
8. Weikl S, Bogenberger K. Relocation Strategies and Algorithms for Free-Floating Car Sharing Systems. IEEE Intell Transp Syst Mag. 2013;5(4):100–11.
9. Kypriadis D, Pantziou G, Konstantopoulos C, Gavalas D. Optimizing Relocation Cost in Free-Floating Car-Sharing Systems. IEEE Trans Intell Transp Syst. 2020;21(9):4017–30. https://doi.org/10.1109/tits.2020.2995197.
10. Laptev N, Yosinski J, Li LE, Smyl S. Time-series extreme event forecasting with neural networks at uber. In: Int Conf Mach Learn; 2017. p. 1–5.
11. Taylor SJ, Letham B. Forecasting at Scale. Am Stat. 2018;72(1):37–45.
12. Papacharalampous G, Tyralis H, Koutsoyiannis D. Predictability of monthly temperature and precipitation using automatic time series forecasting methods. Acta Geophys. 2018;66(4):807–31.
13. Alencar VA, Pessamilio L, da Silva F, Bernardino H, Vieira A. Predição de séries temporais de demanda em modelos de compartilhamento de veículos para modelos uni e multi variáveis. In: Anais do IV Workshop de Computação Urbana. Porto Alegre: SBC; 2020. p. 84–96. https://doi.org/10.5753/courb.2020.12355.
14. Müller J, Bogenberger K. Time Series Analysis of Booking Data of a Free-Floating Carsharing System in Berlin. Elsevier Transp Res Procedia. 2015;10:345–54.
15. Samal K, Babu KS, Das SK, Acharaya A. Time series based air pollution forecasting using sarima and prophet model. In: Proceedings of the 2019 International Conference on Information Technology and Computer Communications. ACM; 2019. p. 80–5. https://doi.org/10.1145/3355402.3355417.
16. EVO API. 2018. https://www.evo.ca/api/Cars.aspx. Accessed 15 July 2018.
17. MODO API. 2018. http://modo.coop/api/, accessed 15 July 2018.
18. Car2Go API. 2018. https://www.car2go.com/api/tou.htm, accessed 31 Jan 2018.
19. World Weather Online. 2020. https://www.worldweatheronline.com, Accessed 18 May 2020.
20. Friedman JH. Greedy function approximation: A gradient boosting machine. Ann Stat. 2001;29(5):1189–232. https://doi.org/10.1214/aos/1013203451.
21. Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM; 2016. p. 785–94. https://doi.org/10.1145/2939672.2939785.
22. Prokhorenkova L, Gusev G, Vorobev A, Dorogush AV, Gulin A. CatBoost: unbiased boosting with categorical features. In: Proc. of the Conference on Neural Information Processing Systems (NeurIPS). Montrèal; 2018. p. 6639–49.
23. Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu T-Y. Lightgbm: A Highly Efficient Gradient Boosting Decision Tree. In: Advances in Neural Information Processing Systems. Long Beach, California; 2017. p. 3149–57.