Forecasting the Carsharing Service Demand Using Uni and Multivariable Models

Victor Aquiles Alencar (✉ fvictoraquiles@ice.ufjf.br)

Universidade Federal de Juiz de Fora

Lucas Ribeiro Pessamilio

Universidade Federal de Juiz de Fora

Felipe Rooke da Silva

Universidade Federal de Juiz de Fora

Heder Soares Bernardino

Universidade Federal de Juiz de Fora

Alex Borges Vieira

Universidade Federal de Juiz de Fora <https://orcid.org/0000-0003-0821-126X>

Research

Keywords: Car-sharing, Prediction, Deep-learning

Posted Date: November 5th, 2020

DOI: <https://doi.org/10.21203/rs.3.rs-99113/v1>

License: This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Version of Record: A version of this preprint was published at Journal of Internet Services and Applications on August 4th, 2021. See the published version at <https://doi.org/10.1186/s13174-021-00137-8>.

RESEARCH

Forecasting the Carsharing Service Demand Using Uni and Multivariable Models

Victor Aquiles Alencar^{*}, Lucas Ribeiro Pessamilio, Felipe Rooke da Silva, Heder Soares Bernardino and Alex Borges Vieira

^{*}Correspondence:

victoraquiles@ice.ufjf.br

Universidade Federal de Juiz de
Fora, Rua Jose Lourenco Kelmer,
36036-900 Juiz de Fora, Brazil

Full list of author information is
available at the end of the article

[†]Equal contributor

Abstract

Carsharing is an alternative to urban mobility that has been widely adopted. However, this approach is prone to several problems, such as fleet imbalance, due to the variance of the daily demand in large urban centers. In this work, we apply two time series techniques, namely, Long Short-Term Memory (LSTM) and Prophet, to infer the demand for three real carsharing services. We also apply several state-of-the-art models on free-floating data in order to get a better understanding of what works best for this type of data. In addition to historical data, we also use climatic attributes in LSTM applications. As a result, the addition of meteorological data improved the model's performance, especially on Evo: an average Mean Absolute Error (MAE) of approximately 61.13 travels was obtained with the demand data on Evo, while MAE equals 32.72 travels was observed when adding the climatic data, the other datasets also improved but none other improved this much. For the free-floating data test, we got the Boosting Algorithms (XGBoost, Catboost, and LightGBM) got the best performance short term, the worst one has an improvement of around 22% of MAE over the next best-ranked (Prophet). Meanwhile in the long term Prophet got the best MAE result, around 22.5% better than the second-best (LSTM).

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, 2, 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. Clear, the increasing complexity interactions of such tasks require intelligent resources for their better 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 [2, 1, 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 more than 22 thousand vehicles in circulation in the Americas [3].

Carsharing services presents three main business models. Two of these models base their services on stations while the reminder 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 left at the origin station. The one-way model allows one to pick up and left a vehicle at any station. Finally, the free-floating model allows one to pick up and left a vehicle at any parking area of the available 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, 5, 2]. In these recent works, researchers have shown results shown that the use of carsharing services follows a diuturnal pattern and, in one of the biggest challenges related to one-way and free-floating carsharing models is the fleet imbalance over the service area. Fleet imbalance occurs when a large portion 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' unsatisfaction [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 the VReP problem: a 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 easily 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 adapt and evaluate time-series prediction models for the carsharing demand. Our work relies on the most popular single and multivariate models, as Prophet [10, 11], LSTM (*Long Short-Term Memory*) [12], and Gradient Boosting [13, 14, 15]. More in deep, we evaluate the prediction models using real data from three carsharing services models located in Vancouver city, Canada. Beyond the carsharing related data, we have also considered external data sources, as the historical meteorological data from the city.

Note that most existing works in this area focus on a unique service model [2, 12], using a specific forecasting technique. In this work, on the other hand, we use several time-series prediction techniques and their variations. Moreover, we consider the main three carsharing service models, using real data. In sum, when compare univariate and multivariate performance, the multivariate model had 10% on average, in the dataset with more missing data. On the short-time prediction, the Gradient Boosting models outperformed the other models, and for a long-turn period, the Prophet obtained the best performance.

The reminder 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. In Sections 5 and 6, we present the results we obtain for each model. 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, 5, 2]. 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*) [12] to forecast the demand for its services. LSTM has been chosen due to its scalability and prediction quality. The modified versions of LSTM in [12] 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 on 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 [16] 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-know 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 [10]. 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 [11]. Other researches used the Prophet model to forecast the Bitcoin value over time, where Prophet performed better than ARIMA [17].

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

We use real-world travel data collected from three carsharing services located 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 datasets contain all trips of the services and their main information, ordered by the time they were collected. In this work, we deal with the time-series of the number of trips per hour, for each service.

More in deep, we have 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. The Evo's website^[1] provides the location of the base stations and the information of available vehicles. The Modo API^[2] returns the vehicles' information from each station, with their booking, available and busy periods. The Car2Go's API^[3] provides the start and end location of each vehicle, and start and end times of the travels. More details regarding the data collection can be obtained in [1].

For multivariate analysis, have obtained meteorological information using an API, called *World Weather Online*^[4], 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. One can observe that the service with the largest amount of data among those considered here is Car2Go, due to its longest collection period. It is also possible to note that even with an equal collection period, Evo has a significantly higher number of trips compared to Modo, due to the characteristics of usage of their services.

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

Table 1 Quantitative summary of the dataset.

The created datasets contain missing data on different dates and periods of time due to interferences in the data collection process (e.g., interruptions by lack of

^[1]<https://www.evo.ca/api/Cars.aspx>, last access: 15/07/2018

^[2]<http://modo.coop/api/>, last access: 15/07/2018

^[3]<https://www.car2go.com/api/tou.htm>, last access: 01/31/2018

^[4]<https://www.worldweatheronline.com>, last access: 05/18/2020

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 was considered for the Car2Go.

For a deeper analysis of Car2Go, we also considered another –longer– period. We used this carsharing service as the dataset of Car2Go is the largest we collected and that one with the smallest number of missing data. We chosen the period from December 13, 2016, to February 25, 2017, to further evaluate Car2Go.

Finally, all the datasets and source codes implemented in this work are public available^[5].

4 Materials and Methods

In this section, we briefly introduce the models and techniques we used here to forecast the carsharing 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 models.

4.1 ARIMA and SARIMA

The Auto-Regressive Integrated Moving Average (ARIMA) and its derivations are widely used in the time-series forecast [16, 2, 17]. ARIMA 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, including 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 autoregressive terms (p), the number of times one has to differentiate data to obtain a stationary data (d), and the number of moving average terms (q). As SARIMA is based on ARIMA, it has the same parameters plus parameters to indicate the seasonality. Particularly, a parameter to indicate the number of periods in each season (m) is considered. We used the pmdarima library^[6] in the 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 has strong seasonality patterns and that suffer effects generated by holidays or extreme events [10]. 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.

^[5]<https://github.com/netlabufjf/timeseries-forecast>, last access: 10/08/2020

^[6]<https://alkaline-ml.com/pmdarima/index.html>, last access: 08/26/2020

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

$$y(t) = g(t) + s(t) + h(t) + e_t,$$

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 [10].

4.3 Gradient Boost

Gradient Boosting is an ensemble of models that produces an improved model by fitting a new predictor with the residual error made by the previous predictor [18]. In this work, 3 gradient boosting models were used: 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 use approximate tree learning [13]. CatBoost introduces the ordered boosting approach that is an alternative to the classical boosting algorithms and processes categorical features [15]. LightGBM is composed by two techniques: the Gradient Based One-Sided Sampling, that exclude small proportions of data that have small gradients; and the Exclusive Feature Bundling, that bundle mutually exclusive features to reduce its number [14].

4.4 LSTM

The *Long Short-Term Memory* (LSTM) is a type of neural network that stands out for its time series prediction [2, 12]. It is based on a Recurrent Neural Network which maintains a short-term memory to influence the forecast of the next time interval. Thus, during the training, it determines the time interval that will be used as memory and the horizon that will be predicted. The data model uses are organized according to the following format: [batch, time stamps, columns]. The lot size indicates the number of examples the method will work on. The number of time steps represents how many timestamps will be contained in each batch 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

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

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 $1/\sqrt{n}$.

Similarly to RMSE, MAE measures the difference between expected and predicted values. The MAE metric is defined as

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

5 Forecasting the Carsharing Demand for Three Real-World Services

In this section, we presented the application of Prophet and LSTM to the carsharing system demand forecasting using both, univariate and multivariate models. For Prophet, which is a univariate model, we have split the dataset into two sets: a training set, with 80% of the data, and a testing set, with the remaining data. 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. The approaches are evaluated according to MAE and RMSE using the test sets.

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

For this analysis, the Prophet parameters were set as the default ones, except by the inclusion of the Canada Day holiday, that occurred on July 1, 2017, and July 2, 2018. The LSTM parameters were chosen by a Grid Search process and can be seen in Table 2.

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, }

Table 2 LSTM parameters summary.

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 (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 present the expected and predicted demands of the carsharing services when using Prophet. The y-axis is the carsharing system demand and the x-axis indicates the instants of time. The carsharing demand values are displayed using points, and the predicted values are represented by the blue and orange curves, where the blue curve is the prediction of the training data and the orange (dashed

line) is the forecast of the first 12 hours of the test set. In addition, the range of uncertainty in the forecasts is displayed along with the lines.

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 for the future 12 hours 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.

Figure 2 shows the forecasting performance of the Prophet for the Modo carsharing system data. Initially, one can observe less missing data in this time series when compared to Evo (Figure 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 for the testing data (12 hours) returns the morning peak, which decays when entering the afternoon. In the two-way service, the usage behavior follows the idea of occupying vehicles in the morning for use along the day, which justifies the peak in the morning and another lower peak at noon. It is important to notice that Figure 2 shows a point with high value that could be considered as an outlier. This point was not removed 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 services. First, one can see that the model's behavior is similar to Evo (Figure 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 12-hours forecast reflected the constant growth in demand that starts in the morning and reaches a global 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 obtained when using the univariate and multivariate LSTM models. Figures 4-6 show the real values and predicted curves for the testing data (12h). For each carsharing service, one can observe that both univariate and multivariate models present similar patterns.

Figure 4 shows the results of the univariate and multivariate LSTM models for the Evo carsharing service. In both cases, the univariate and the multivariate LSTM models follow the actual service demand. However, the univariate model 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 is able to 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 Figures 6, the univariate and the multivariate LSTM models present similar performance for this carsharing service and, visually, both models present good results.

5.3 Summary and discussion

Metric	Dataset	Uni LSTM	Multi LSTM	Prophet
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

Table 3 Values obtained by the models analyzed here.

Table 3 shown RMSE and MAE for the three services evaluated here, and the mean and standard deviation values of the predicted demands. The Prophet results in the table 3, shows that the models performed well for the RMSE, where low values represent how well the forecast fits the curve shape. This good performance can be noticed with the Evo, which has more missing data that can harm the training of the model but yet obtained the best values among the other models. For the MAE values, we can notice that the error related to the values of the Modo and Car2Go is low, indicating the reliability over them. However, the values obtained by Evo are higher than the others, again due to the missing data.

The univariate LSTM model results show that the RMSE indicates a good performance for Car2Go and Modo, confirming what we visually testified before. For Evo, the LSTM is, at least 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 of the multivariate LSTM model are very close to the values of the univariate model presented in the table 3, except for Car2Go 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 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 Demand Forecasting of the Free-floating Carsharing Model

In this section, we further investigate and evaluate the free-floating carsharing system demand. We predict service demand using multiple state-of-art models and techniques.

Two groups of models are evaluated here: (i) the statistical models, namely, ARIMA, SARIMA, and Prophet; and (ii) the multivariate models, namely Gradient Boost and LSTM. The data was split into two sets, namely, training and test sets, with 80% and 20% of the original dataset, respectively. The analysis is conducted using the test set.

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 turn future. We have chosen a 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 turn periods. We have chosen a week period due to the seasonality of the demand (i.e., the same weekday tend 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 ARIMA and SARIMA parameters are $(p, q, d) = (2, 0, 3)$ and $(p, q, d)(P, D, Q)_m = (2, 0, 3)(2, 1, 1)_{12}$ respectively. 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^[7]. Table 5 presents the parameters used here for Gradient Boost. More details on the libraries used for the experiments with Gradient Boost methods can be found in their respective documentations^{[8],[9],[10]}.

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

Table 4 Prophet 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 }

Table 5 Gradient Boost parameters.

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

^[7]https://facebook.github.io/prophet/docs/quick_start.html, last access: 10/08/2020

^[8]<https://xgboost.readthedocs.io/en/latest/parameter.html>, last access: 10/08/2020

^[9]https://catboost.ai/docs/concepts/python-reference_parameters-list.html, last access: 10/08/2020

^[10]<https://lightgbm.readthedocs.io/en/latest/Parameters.html>, last access: 10/08/2020

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 turn forecasting), the LSTM models use two layers with 220 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 12 nodes. For the model that forecasts the next week (i.e., long turn 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 time 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 its 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 Turn Forecasting

Figures 7-13 present a visual example of the short turn carsharing demand forecasting. Each figure refers to a given model, as ARIMA (Figure 7), SARIMA (Figure 8), Prophet (Figure 9), CatBosst (Figure 10), LightGBM (Figure 11), XGBoost (Figure 12), and LSTM (Figure 13). In these figures, we present a 12 hour period where the dots represents the expected values and the line the predicted values.

First, one can notice that the ARIMA and SARIMA models (Figures 7 and 8) do not show a good performance to predict the carsharing system demand, as the predicted data does 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 (Figure 9). All Gradient Boosting models performed similarly (Figures 10-12). These models fit well the shape of the real data, with fewer errors for the first 6 hours. Finally, LSTM also fits well the shape of the expected values. However, when compared to the Gradient Boosting models, LSTM presents larger errors during the initial period of prediction.

Figure 14 shows the MAE values for all the test set, and all models we evaluate. The performance of ARIMA was not included in this figure due to visualization issues. Moreover, ARIMA presents the worst performance among the models analyzed here and an MAE equals 94.451198. According to Figure 14, CatBosst, 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 values. 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.

	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

Table 6 Values obtained by the techniques used here when predicting the carsharing demand at short turn (12 hours).

6.2 Long Turn Forecasting

As in Section 6.1, where we presented the short turn period forecasting of the carsharing system demand, Figures 15-21 show the long turn forecasting results. As shown in Figure 15, ARIMA poorly performs to forecast the long turn period. On the other hand, SARIMA performs well, especially for the first half of the period considered here. Prophet (Figure 17) performs well for the entire period. Indeed, this model fits well the curve for almost all the long-turn period.

Figure 9 12-hours prediction of Prophet.

Figure 10 12-hours prediction of CatBoost.

Differently to the short-turn periods forecasting, the Gradient Boosting models degenerated quickly. As shown in Figures 18-20, the Gradient Boosting methods do not performed well after –about– 75 hours. According to Figure 21, LSTM performed similarly to Prophet. However, LSTM does not found good predictions for the initial 24 hours.

Figure 22 shows the MAE values for the models we evaluate in this work. Again, as occurred to the short-turn 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 performance. Prophet obtained the best demand predictions for the next 168 hours when compared to the methods tested here.

Table 7 presents the MAE and RMSE of the predictions, and the mean and standard deviation values (for the forecasted demand). Prophet obtained the best overall performance. The Gradient Boosting techniques and Arima found the worst

Figure 11 12-hours prediction of LightGBM.

Figure 12 12-hours prediction of XGBoost.

results. Sarima performed similarly to Catboost. LSTM obtained the second-best performance for the long-turn predictions (one week), differently from the results of the short-turn case (12h).

	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

Table 7 Values obtained by the techniques used here when predicting the carsharing demand at long-turn (one week).

7 Conclusion

In this article, we propose the demand predictive models for three carsharing services, and we further evaluate and analyze predictive models for a free-floating service. The time-series used here was obtained from services from Vancouver, Canada, and the dataset was filtered to reducing missing data. In the first scenario, three models were proposed: two univariate and one multivariate. On the other hand, 7 models were proposed for the second scenario.

Figure 13 12-hours prediction of LSTM.

Figure 14 Model errors for the short-turn (12-hours) forecasting.

The analysis of the models in the first scenario (using data of three carsharing services) showed that all predicting techniques obtained a similar performance, even when the dataset contains missing data. The values predicted by Prophet fitted well the expected ones when missing data exists. Also, Prophet outperformed the other models when using a data set with less missing data, as MODO set. The Univariate LSTM model presented results similar to those found by Prophet, but Prophet obtained better results on at least two datasets. The Multivariate LSTM considers meteorological data also and, as a consequence, performed better than the univariate models. The results found by the Multivariate LSTM were better but similar to those obtained by Prophet. The exception is when using the MODO dataset, where Prophet achieved the best results.

In the second scenario, the statistical models ARIMA and SARIMA obtained the worst predictions. On the other hand, the Prophet obtained the best performance when forecasting long-turn data (one week). In short-turn cases, the best models were the Gradient Boosting models, with similar performances, followed by LSTM. However, the Gradient Boosting methods do not have the same performance on the long time prediction. In this last case, SARIMA performed better than the Gradient Boosting techniques.

As future work, geolocation characteristics can be included in the dataset to determine the demand by service operation zones and to provide more specific operational demands. Also, the forecast models can be used with optimization methods

Figure 15 One week prediction of ARIMA.

Figure 16 One week prediction of SARIMA.

to support the decision making procedure. Finally, we expect that this research improves the forecasting and monitoring strategies for vehicle sharing and other similar services.

8 List of 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

Ethics approval and consent to participate

Not applicable

Consent for publication

Not applicable

Availability of data and material

Please contact author for data requests.

Funding

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

Authors' contributions

All authors read and approved the final manuscript.

Acknowledgements

Authors thanks the support of Capes, CNPq and Fapemig.

Competing Interests

The authors declare that they have no competing interests.

Figure 17 One week prediction of Prophet.

Figure 18 One week prediction of LightGBM.

References

- Alencar, V.A., Rooke, F., Cocca, M., Vassio, L., Almeida, J., Vieira, A.B.: Characterizing Client Usage Patterns and Service Demand for Car-Sharing Systems. *Information Systems*, 101448 (2019)
- Cocca, M., Teixeira, D., Vassio, L., Mellia, M., Almeida, J.M., Couto da Silva, A.P.: On Car-Sharing Usage Prediction with Open Socio-Demographic Data. *Electronics* **9**(1), 72 (2020)
- Shaheen, S.A.: Mobility and the Sharing Economy. *Transport Policy* **51**(Supplement C), 141–142 (2016). doi:10.1016/j.tranpol.2016.01.008
- Hamari, J., Sjöklint, M., Ukkonen, A.: The Sharing Economy: Why People Participate in Collaborative Consumption. *Journal of the Association for Information Science and Technology* **67**(9), 2047–2059 (2016)
- 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)*, pp. 572–577 (2016)
- Nourinejad, M.: *Dynamic Optimization Models for Ridesharing and Carsharing*. Master's thesis, University of Toronto (March 2014)
- Illgen, S., Höck, M.: Literature Review of the Vehicle Relocation Problem in One-Way Car Sharing Networks. *Transportation Research Part B: Methodological* **120**, 193–204 (2019)
- Weikl, S., Bogenberger, K.: Relocation Strategies and Algorithms for Free-Floating Car Sharing Systems. *IEEE Intelligent Transportation Systems Magazine* **5**(4), 100–111 (2013)
- Kypriadi, D., Pantziou, G., Konstantopoulos, C., Gavalas, D.: Optimizing Relocation Cost in Free-Floating Car-Sharing Systems. *IEEE Transactions on Intelligent Transportation Systems* (2020)
- Taylor, S.J., Letham, B.: Forecasting at Scale. *The American Statistician* **72**(1), 37–45 (2018)
- Papacharalampous, G., Tyralis, H., Koutsoyiannis, D.: Predictability of monthly temperature and precipitation using automatic time series forecasting methods. *Acta Geophysica* **66**(4), 807–831 (2018)
- Laptev, N., Yosinski, J., Li, L.E., Smyl, S.: Time-series extreme event forecasting with neural networks at uber. In: *International Conference on Machine Learning*, vol. 34, pp. 1–5 (2017)
- 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*, pp. 785–794 (2016)
- 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*, pp. 3146–3154 (2017)
- Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A.V., Gulin, A.: CatBoost: Unbiased Boosting with Categorical Features. In: *Advances in Neural Information Processing Systems*, pp. 6638–6648 (2018)

Figure 19 One week prediction of XGBoost.

Figure 20 One week prediction of CatBoost.

16. Müller, J., Bogenberger, K.: Time Series Analysis of Booking Data of a Free-Floating Carsharing System in Berlin. Elsevier, Transportation Research Procedia **10**, 345–354 (2015)
17. Samal, K., Babu, K.S., Das, S.K., 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, pp. 80–85 (2019). ACM
18. Friedman, J.H.: Greedy Function Approximation: a Gradient Boosting Machine. Annals of Statistics, 1189–1232 (2001)

Figures

Figure 1

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

Figure 2

12-hours forecast by Prophet model for MODO.

Figure 3

12-hours forecast by Prophet model for Car2Go.

(a) Univariate

(b) Multivariate

Figure 4

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

(a) Univariate

(b) Multivariate

Figure 5

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

(a) Univariate

(b) Multivariate

Figure 6

12-hours forecast using the LSTM model for Car2Go.

Figure 7

12-hours prediction of ARIMA.

Figure 8

12-hours prediction of SARIMA.

Figure 9

12-hours prediction of Prophet.

Figure 10

12-hours prediction of CatBoost.

Figure 11

12-hours prediction of LightGBM.

Figure 12

12-hours prediction of XGBoost.

Figure 13

12-hours prediction of LSTM.

Figure 14

Model errors for the short-turn (12-hours) forecasting.

Figure 15

One week prediction of ARIMA.

Figure 16

One week prediction of SARIMA.

Figure 17

One week prediction of Prophet.

Figure 18

One week prediction of LightGBM.

Figure 19

One week prediction of XGBoost.

Figure 20

One week prediction of CatBoost.

Figure 21

One week prediction of LSTM.

Figure 22

Model Errors for the long-turn (one-week) forecasting.