Peer-reviewed version available at *Electronics* **2020**, *9*, 72; doi:10.3390/electronics9010072

Article

On Car Sharing Usage Prediction with Open Socio-demographic Data

Michele Cocca[2]*, Douglas Teixeira[1], Luca Vassio[2], Marco Mellia[2], Jussara M. Almeida[1], and Ana Paula Couto da Silva[1]

[1] Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, Brazil; {douglas,jussara,ana.coutosilva}@dcc.ufmg.br

[2] Department of Electronics and Telecommunications, Politecnico di Torino, Italy; {michele.cocca,luca.vassio,marco.mellia}@polito.it

* Correspondence: michele.cocca@polito.it

Abstract: Free Floating Car Sharing (FFCS) services are a flexible alternative to car ownership. These transportation services show highly dynamic usage both over different hours of the day, and across different city areas. In this work, we study the problem of predicting FFCS demand patterns – a problem of great importance to an adequate provisioning of the service. We tackle both the prediction of the demand i) over time and ii) over space. We rely on months of real FFCS rides in Vancouver, which constitute our ground truth. We enrich this data with detailed socio-demographic information obtained from large open-data repositories to predict usage patterns. Our aim is to offer a thorough comparison of several machine learning algorithms in terms of accuracy and easiness of training, and to assess the effectiveness of current state-of-art approaches to address the prediction problem. Our results show that it is possible to predict the future usage with relative errors down to 10%, and the spatial prediction can be estimated with relative errors of about 40%. Our study also uncovered the socio-demographic features that most strongly correlate with FFCS usage, providing interesting insights for providers opening service in new regions.

1. Introduction

Transportation in urban areas is among the top challenges to improve people quality of life and to reduce pollution. Historically, private vehicles have been the preferred transport means, with municipalities investing in public transportation systems to offer alternatives to reduce traffic and pollution. With the birth of the sharing economy, we are now assisting to a transition towards new forms of shared mobility, which have spurred the interest of both the research community and the private companies willing to start new businesses.

Car sharing is an evolution of the classic car rental model where users can rent cars on-demand for a short period, e.g., a 20-minutes trip across town. In particular, Free Floating Car Sharing (FFCS) services let customers pick and return the cars everywhere inside a city. Customers reserve, unlock and return the car by using an application on their smartphones. One such service is Car2go [1], which currently operates in several cities in the world. In the FFCS implementation, the provider bills the user only for the time spent driving, with simple minute-based fares which factors all costs. Some studies demonstrated that a massive adoption of car sharing service can improve mobility quality and reduce cost and pollution (see for instance [1], [2] and [3]).

[1] https://www.car2go.com/

To properly design and manage a FFCS service, a provider needs to know which is the demand of mobility over the different period of the day, and over the different areas of the city. The prediction of FFCS demand patterns is thus fundamental for an adequate provisioning of the service. Armed with good predictions, the provider can better plan long term system management, e.g., whether to extend the operative area to those neighborhoods with expected customer growth. Similarly, it can implement short term dynamic relocation policies to better meet the next hours demand [4–6].

In this work we investigate the dynamics of usage of a real FFCS service. We aim at assessing how state-of-the-art machine learning algorithms can help FFCS providers and policy makers in predicting the demand, both over time and across different spatial regions. In more details we leverage a dataset of real rides from cities where Car2go is offering its FFCS service. We take as case study the city of Vancouver, Canada, the city with highest demand in our dataset. We rely on more than 1 million rentals covering 9 months in 2017 [7]. We augment the dataset by exploiting a rich and heterogeneous open dataset, namely the 2016 Vancouver Municipality census.[2] This second dataset comprises more than 800 features, which span from detailed information about shops in each neighborhood to weather conditions, from information about residents to rate of emergency calls throughout the day. Our aim is to first assess to which extent it is possible to predict the FFCS demand, and second, which of this data has a higher prediction power.

We focus on two scenarios: in the first one we investigate how to predict the demand in the future, only considering only past usage and weather conditions. This is fundamental for managing the FFCS fleet both in the short term (e.g., implementing relocation policies during service peak time), and in the long-term (e.g., to properly match the fleet size to the future system growth). To this end, we analyse machine learning algorithms that are considered state of art, from simple Linear Regression and traditional Seasonal Auto Regressive Integrated Moving Average (SARIMA) models, to Random Forests Regression (RFR), Support Vector Regression (SVR) and latest approaches based on Long Short-Term Memory Neural Networks (NN) [8,9]. With their increasing complexity, we aim at assessing not only how they perform in our target prediction task, but also to which extent one would need to embrace a complex model (such as NN) or rather simpler and more informative models (like linear regression and RFR).

In the second scenario, we correlate socio-demographic indicators with FFCS demand. We predict the demand of cars in a neighborhood without past data, using only socio-demographic data. This problem is often referred to as a green field, or cold start, approach. In this case, and the operator is interested in knowing which could be the possible system usage in a new neighborhood (or even a new city) based only on socio-demographic data. We map the FFCS demand to Vancouver neighborhoods, and associate them to the socio-demographic data coming from the official Vancouver census. We then use again machine learning techniques to highlight the relationship between demographics and customers' mobility. We aim at answering the following research questions: i) Using modern machine learning methodologies, and armed with a rich socio-demographic data, would one be able to predict the mobility pattern in a city? And ii) which would be the most important socio-demographic data to use for this task?

Through a series of thorough experiments, we show that the temporal prediction of rentals can be solved with errors down to 10% when using modern machine learning algorithms. Interestingly, Random Forests turn out to perform stably better than the other models, including Neural Networks, for this task. When considering the mobility prediction using socio-demographic data only, we obtain errors in the 40-50% range. While possibly not accurate enough for a precise planning, this prediction still would be useful for operators willing to decide, e.g., to which new areas of the city to extend their service. Interestingly, our models allow us also to observe which features are the most useful for the prediction problem, a precious information for providers and regulators to understand FFCS systems,

[2] https://opendata.vancouver.ca/pages/home/

for instance to decide in which new cities to start a new service (green field problem). For example, results show that the density of people commuting by walk and the number of emergency calls in a neighborhood are important factors for predicting the number of rentals that will start there. Instead, for the temporal prediction, knowing the weather conditions in the near future would help.

After overviewing the related work in Sec. 2, we describe the data collection methodology we adopt in Sec. 3. Sec. 4 provides a characterization of the datasets, while Sec. 5 and Sec. 6 provide details about the methodologies we uses, and results for the temporal and spatial prediction, respectively. Finally, Sec. 7 summarizes our findings.

2. Related work

With the easiness of collecting data, and the ability to build and train machine learning solutions, researchers have started applying data driven approaches in the context of transportation. Authors of [10] are among the first to address the traffic modelling and prediction with real traffic data. The authors improved congestion prediction algorithm using a Kalman filtering approach, showing how traffic is stationary in time. Later, authors of [11] proposed a new approach based on a multivariate extension of non-parametric regression to predict traffic patterns, aiming at counteracting traffic congestion. While similar in spirit, our work focuses on FFCS services explicitly, and uses a much richer dataset as well as more advanced machine learning algorithms.

Focusing on car sharing, early work focused on estimating demand using activity-based micro-simulation to model how agents move around in a city [12]. Later on, as data from operative car sharing platforms became available, researchers started using real data to analyze mobility demand. Authors of [13] proposed a demand model to forecast the modal split of the urban transport demand. Similarly, authors of [14] investigated the Mobility-as-a-Service market opportunities, where FFCS is one of the implementations, and pointed out how FFCS supply can push the users to avoid a new car purchase, leading to a reduction of CO_2 emission [2]. Yet none of these prior studies focused on demand prediction. Similar in spirit, authors of [15] made a large survey covering a Swiss station-based car sharing service. The results confirmed that FFCS is preferred as a fast alternative to public transportation and the subscription depends on the different car sharing implementation. Complementarily, authors of [4] proposed a simple binary logistic model for predicting car sharing subscribers in Switzerland, considering the relationship between potential membership and service availability. Then, the authors used this prediction to locate unmet demand areas where to place a new car sharing station. Similarly, authors of [16,17] conducted a detailed characterization of a car sharing system in Munich and Berlin. Similarly to our work, they identified features correlated with the demand for shared cars in the analyzed cities. We here analyze a much larger set of features, including demographics and economic data, and consider multiple prediction models. We focus on demand prediction, facing both time and space dimensions, and provide a thorough comparison and guidelines for future directions.

In our previous work [18] we analyzed in depth the usage of different car sharing systems in Vancouver. Based on this data we developed a model of FFCS usage and built a simulator to design new systems based on electric vehicles [5]. In particular we tackled the charging station placement problem, showing that the optimal placement requires few stations to satisfy charging requests in different cities [6].

To the best of our knowledge, we are the first to face the FFCS demand prediction tackling both the temporal and spatial prediction. Moreover, we are the first to use a very large dataset including dozens of features to solve this problem. This let us provide also detailed insights on which of those features are the most important ones to solve our target prediction problem.

3. Data gathering methodology

3.1. FFCS data collection

We collect data from Car2go, a popular FFCS system offering service in more than 25 cities, with more than 25 millions rentals worldwide, and 3.6 millions customers. The Car2go service collects and shares the position of all cars in its fleet. A customer looks for and reserves a car by using a smartphone application. At the end of the ride, the customer parks and returns the car by notifying the FFCS system, using again the smartphone application. The backend system records the new position of the car, and makes it available for other customers. Car2go allows developers to interact with their services through a public Application Programming Interfaces (API).[3] With these APIs, we can retrieve the current position of available cars in a given city. Each car is identified by its plate, so that it is possible to identify rentals by simply performing periodic queries. In our past work [7], we engineered UMAP – Urban Mobility Analysis Platform – which allows us to systematically collect precise data about car rentals in all cities where Car2go offers a service.

In details, UMAP queries the Car2go API every minute to get the currently available cars. It then rebuilds the history of rentals of each car, identifying *bookings* and *parkings*. A *booking* is the time period in which a car is booked by a customer (or in maintenance). Conversely, a *parking* is the time period during which a car has been available for a ride to users. Since customers can reserve a car and then cancel the reservation afterwards without actually renting it, we consider a *rental* a booking having (i) distance between starting and final locations greater than 500 m; (ii) travel duration shorter than 1 hour. In a nutshell, we discard those bookings which where not converted into rentals (i.e., when the user reserved the car without actually driving it), and those rentals where the car disappears for long periods (i.e., possibly due to maintenance). We refer the reader to [7] for a detailed analysis of these thresholds.

Here we focus on Car2go rentals recorded in Vancouver, during 9 months of 2017. We chose Vancouver among the cities where Car2Go offers a service because of the high availability and richness of open data directly made available by its municipality, as discussed in the next section. In total, we collected more than 1 million rentals that we use as ground truth to train and test machine learning based algorithms to predict service demand.

3.2. Socio-demographic, weather and other open data

We also explore socio-demographic data that can be used as external inputs to car usage prediction algorithms. Specifically we consider Vancouver census open data.[4] We use the Vancouver official neighborhood definition which identifies 22 neighborhoods. Per each neighborhood, the census dataset provides detailed socio-demographic information such as number of residents in a given range of age, with a certain income, household compositions and commuting habits. For each neighborhood the census reports also information about services that are located in it, e.g., shops, bus stops and parking places. In total, the census presents more than 800 socio-demographic and other spatial features. Among those, we manually selected 83 features that might be related to human mobility.[5] Moreover, we also report: i) the distance to downtown – computed as the distance from the neighborhood to the downtown neighborhood (considered as the central area);[6] ii) an indicator of human activity, measured by the number of emergency calls per time bin (obtained from the Vancouver census); and iii) the

[3] The use of the Car2go API (https://www.car2go.com/api/tou.htm) is subject to approval by Car2go. We got the approval in September 2016 and continued the collection of data in January 2018.
[4] https://opendata.vancouver.ca/pages/home/
[5] The list of features is available at https://opendata.vancouver.ca/pages/census-local-area-profiles-2016-attributes/
[6] We use the neighborhoods central points for distance computation.

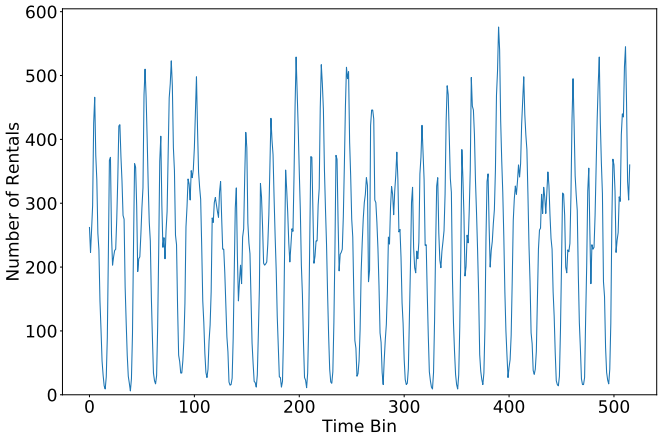


Figure 1. Time series of starting rentals in September 2017 aggregated per hour

hourly weather for Vancouver – as directly available from the OpenWeather project.[7] For each of the 22 neighborhoods, we normalize each numerical feature by the neighborhood total area.

Our aim is to include a superset of features possibly correlated with human mobility and thus car rental prediction, so as to provide the machine learning algorithms with an input dataset as rich and diverse as possible to learn from.

4. Dataset overview

We first provide an overview of the data at our disposal offering insights into the diversity and heterogeneity present both in the temporal and spatial FFCS usage patterns as well as in the socio-demographic data.

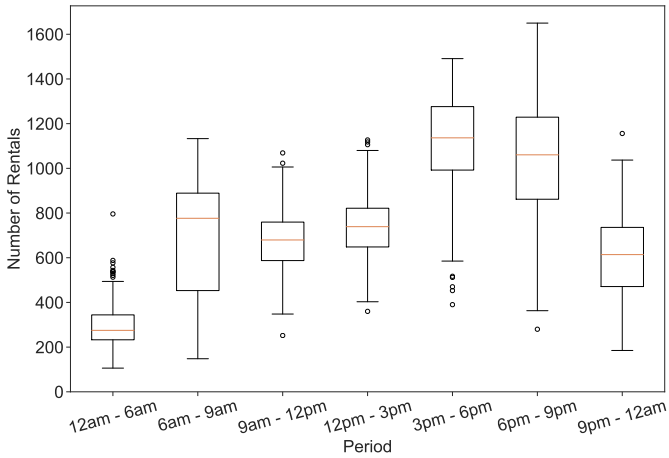
4.1. FFCS temporal characterization

We start by showing the temporal evolution of rentals over time. Figure 1 shows the total number of starting rentals per hour in the whole city during September 2017. Even if we can spot some periodicity, there is a lot of variability that makes the prediction problem not straightforward. For our analyses, from now on we aggregate rentals both in time and in space. Specifically, given a neighborhood we consider the fraction of rentals *starting* and *ending* there. We aggregate the time series of rentals into 7 time bins per each day, namely from midnight to 6am (night period), and then every 3 hours. This time granularity is typically used for system design and control [17]. The rationale is to provide the FFCS company that actionable information on the demand for cars, e.g., to schedule car maintenance or implement relocation policies. A one-hour period is often too short for the company to be able to respond to changes in demand.

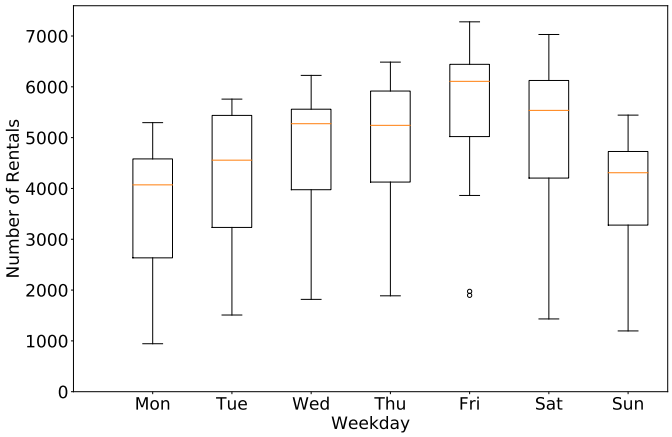
To give more details about the variability of the data, Figure 2a shows boxplots of the numbers of rentals starting in each time bin. Each boxplot represents the quartiles of the distribution, with outliers shown as points .[8] The series shows large variability, with peaks during early mornings (6am-9am) and afternoon (3pm-6pm and 6pm-9pm), and with low values during nighttime (12am-6am). Figure 2b shows boxplots of the total numbers of rentals grouped per day of the week. The number of rentals peaks on Fridays, with significantly lower values registered on Sundays and Mondays. Again, we observe a quite sizeable variability over the days, as observed by in the sizes of the boxplots. Such

[7] https://openweathermap.org/history-bulk

[8] We consider as outliers measures that are outside the mean $\pm$ 2.698 times the standard deviation range.

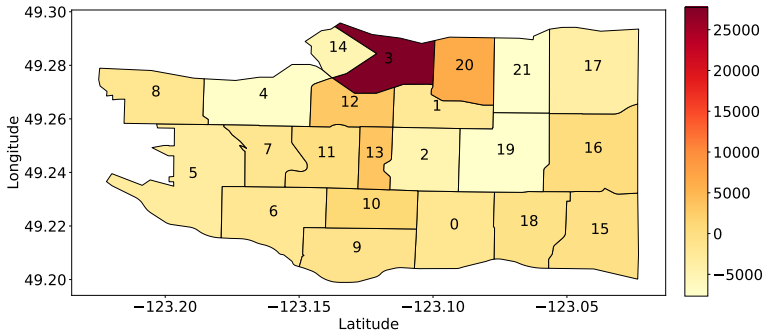


(a) Boxplots of number of rentals starting in each time bin

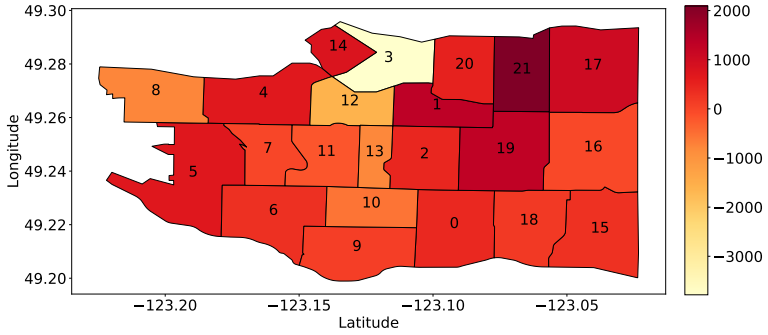


(b) Boxplots of number of rentals starting in each day of the week

Figure 2. Temporal characterization of number of rentals. Boxplots highlighting the variability over the day for the same time bin of the day (top plot), and over different days (bottom plots)



(a) 6am - 9am rental net flow



(b) 6pm - 9pm rental net flow

Figure 3. Heatmap of net flow for each neighborhood in Vancouver. The more the area is red, the higher are the arrivals with respect to the departures. Neighborhood numbering is shown (from 0 to 21)

variability in the number of rentals hints at the fact that prediction models have to be able to deal with sizeable temporal variations in the demand for cars.

4.2. FFCS spatial characterization

We now take a closer look into how these numbers vary across different areas of the city. Rather than providing a complete characterization of the origin/destination matrix (which is outside present scope), we here focus on particular examples to showcase the spatial variability in the demand of mobility. We focus on the morning and afternoon peak time bins (6am-9am and 6pm-9pm). For each neighborhood we compute the *net flow* defined as the difference between the number of rentals starting from that neighborhood, and the number of rentals arriving to that neighborhood during the specified time period. We consider the cumulative net flow in September 2017. Figure 3 depicts the results with a heat map. Darker red neighborhoods means that arrivals exceed departures, i.e., the neighborhood is attracting vehicles. Conversely, lighter colors imply that more vehicles are departing from that neighborhood than arriving in it. Numbers identify different neighborhoods. Downtown business area (number 3) attracts a lot of rides in the morning period (Figure 3a), while the opposite pattern is seen during the afternoon period (Figure 3b). In general, we can assert that the FFCS demand is higher in the peak hours towards downtown in the morning and residential areas in the afternoon, while sensibly lower at nighttime. This is clearly visible in Figure 4 which reports the net flow for two neighborhoods for each hour of the day, namely the downtown neighborhood (number 3), and the Grandview-Woodland (number 21) neighborhood, a residential area close to downtown.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2019 doi:10.20944/preprints201911.0149.v1

Peer-reviewed version available at Electronics 2020, 9, 72; doi:10.3390/electronics9010072

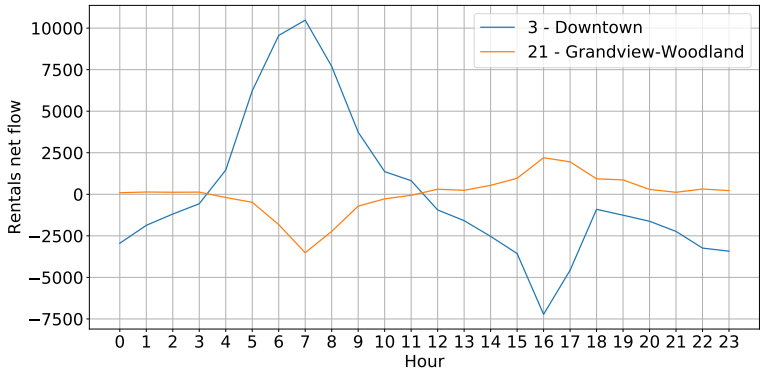


Figure 4. Total net flow in September 2017 for Downtown (neighborhood 3) and Grandview-Woodland (neighborhood 21) over different hours of the day

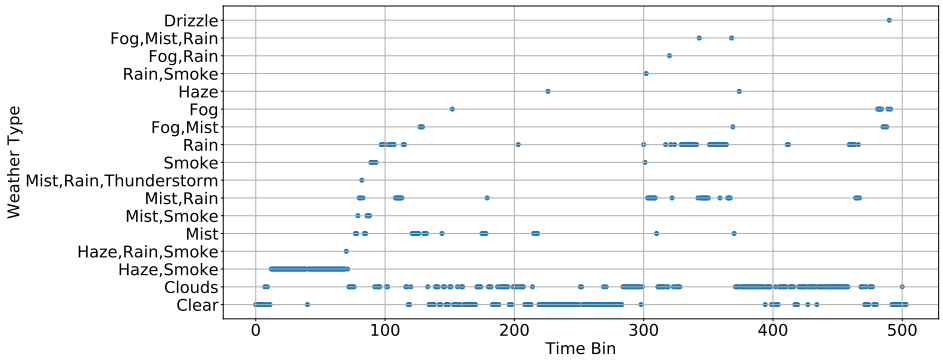


Figure 5. Time series of weather conditions per hour during September 2017. Each point in the plot represents an occurred weather type

4.3. Socio-demographic and weather data characterisation

We now provide some examples of the socio-demographic and open data. Figure 5 reports the weather condition during the month of September 2017. Being it a categorical variable, we assign each weather condition combinations to a different value on the y-axis. As expected, the weather conditions change over time quite frequently. Moreover, comparing the weather conditions with the number of rentals in Figure 1, it is hard to find any evident correlation.

Similarly, Figures 6a and 6b show the number of high-income households and the number of emergency calls per day for each neighborhood, respectively. Also in this case, it is hard to see any clear correlation with the net flow per neighborhood reported in Figures 3. The scenario is similar considering other socio-demographic features.

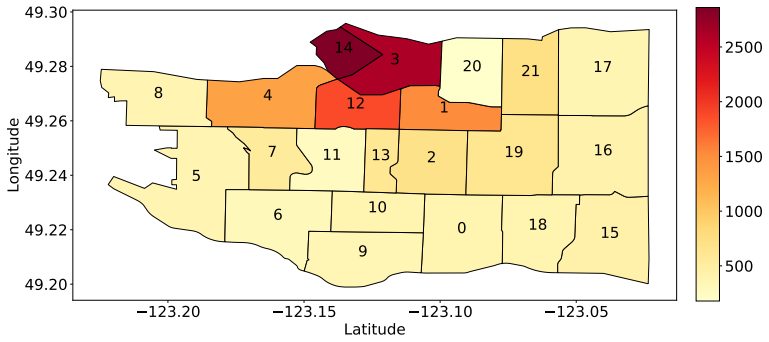
Despite the not linear correlations between the socio-demographic data and rentals, it is possible that the combination of multiple features help the prediction of car rentals, as we will discuss in the next sections. This is exactly what the machine learning algorithms aim at, i.e., building a model from data, leveraging correlation from multiple variables that, considered together, carry enough information to predict system usage. In a nutshell, we let eventually the machine learning model to decide if and how to factor different features in the prediction model.

5. Temporal predictions of rentals

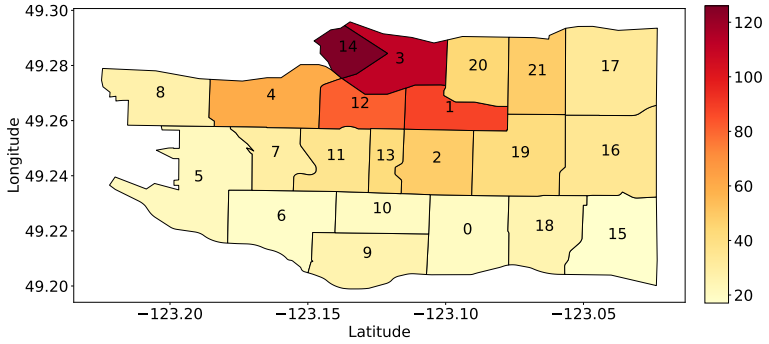
In this section, we describe our task of predicting the number of rentals in the whole city at a target time in the future. Eventually, the same methodology could be applied for each neighborhood. This prediction can exploit historical data, i.e., given the time series of rentals in the past, predict the number of rentals in the future. If only the past time series are available, the problem falls in the univariate regression class, i.e., the prediction is based only on past data of the same target variable.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2019 doi:10.20944/preprints201911.0149.v1

Peer-reviewed version available at Electronics 2020, 9, 72; doi:10.3390/electronics9010072



(a) Maximal income range density per neighborhood



(b) Emergency calls per day per neighborhood

Figure 6. Heatmap of a sample of demographic (top) and socio-demographic (bottom) data at our disposals. These two samples looks quite correlated

Let $x(t)$ be our target variable, i.e., the number of rentals at time t. In the case of prediction with historical data, we predict

$$x(t+j) = f(x(t), x(t-1), \ldots, x(t-k)), j > 0$$

as a function $f()$ of the past $k+1$ data of x itself, where j is the horizon of the prediction.

If we also have other information, we can build a more generic model to consider the dependence to other variables. We want to predict

$$x(t+j) = g(y_1, y_2, \ldots, y_l), j > 0,$$

where $\{y_i\}$ are different variables, possibly a time series themselves (including x) and g is the model that allows us to predict x at time $t+j$. This problem is a multivariate regression problem, where multiple features are used to predict the target variable x.

Considering the time horizon of the prediction, we can formulate two versions of the problems: predict the long-term or short-term usage. In the first case, we build and train a single model using all data at our disposal to predict the system usage in the next months. In the short-term we target the prediction of the next time bin $t+1$ only, i.e., $j = 1$. In this second case, we build and update a new model at each time bin by adding the latest recorded number of rentals to the train set as soon as it becomes available.

Both predictions are important for the car sharing provider. For instance, the long-term predictions are important for the company to know if their fleet size is enough to keep up with the expected demand. The short-term is important for the company to know when to take a car down for maintenance, or when and where cars should be eventually relocated to those neighborhoods where the demand is expected to increase shortly. While for long-term prediction we use the time series of the rentals and

information about day of the week and hour of the day, for short prediction we can use also the near future weather condition information.

In this work, we consider discrete time, i.e., we split time into fixed size time intervals as defined in the aggregation step – see Section 4. We then build and train several machine learning models to tackle each aforementioned problem. Our goal is to compare them in terms of accuracy of the prediction and complexity of the model. At last, we are also interested in considering models that are interpretable, i.e., that allow us to understand which are the most important features that affect car sharing usage in large cities. We evaluate all models considering MAPE (mean absolute percentage error) over the validation set, which is defined as

$$MAPE = \frac{1}{|V|} \sum_{t_i \in V} \frac{| x(t_i) - \hat{x}(t_i) |}{x(t_i)},$$

where V is the validation set, $x(t_i)$ is the actual value of the data at moment t_i and $\hat{x}(t_i)$ is the predicted value.

5.1. Prediction models

We use off-the-shelf machine learning models both for the long-term and short-term scenarios. We evaluate univariate models: a simple baseline (BL) approach, the autoregressive moving average (ARIMA) and the seasonal autoregressive moving average (SARIMA) algorithms. Univariate models do not account for the influence of other time-variant factors such as weather conditions. To account for that, we also investigate the performance of linear regression, Random Forests Regression (RFR), Support Vector Regression (SVR), and long-term short-term memory neural networks (NN). With these algorithms we include categorical features (the day of the week and weather, for instance). Following correct practices [19], we represent each categorical feature as many binary variables, one for each category. For example, when representing a given weather type, the corresponding binary variable will be set to *True* while all the other weather-related variables to *False*. We used the algorithms implementation in Python libraries `scikit-learn`[9] [20] and Keras[10]. Our code for the analysis is publicly available[11]. For details about each model, we refer the reader to [9]. Below we offer a high level description. For each model, we perform a hyper-parameter optimization, not reported here for the sake of brevity. Below we provide the final parameters we use for our experiments.

Baseline. A simple approach to determine $x(t+j)$ in a time bin is to take the average number of rentals in the same time bins in the available past days. We compare all our prediction models to this baseline.

ARIMA. ARIMA (autoregressive integrated moving average) is widely used to predict time series data. ARIMA models are a combination of autoregressive models with moving average models. The creation of an ARIMA model involves specifying three parameters (p, d, q). The d parameter measures how many times we have to differentiate the data to obtain stationary data. After determining d, we use sample partial auto correlation function to get the value p. Finally, we determine the order q by looking at the sample auto correlation function of the differentiated data. We experimented with several combinations of values for the parameters (p, d, q) and the combination that gave us the best results is $(p, d, q) = (2, 0, 1)$.

SARIMA. A SARIMA model incorporates the known seasonality (periodicity) of the data into an ARIMA model, enhancing its predictive power. For instance, when modeling a time series, it is often the case that the data has a daily, weekly, or monthly periodicity. We used our previous ARIMA model

[9] https://scikit-learn.org/
[10] https://keras.io/
[11] https://github.com/dougct/carsharing-prediction

Prediction Model	MAPE [%] Average	MAPE [%] Standard Deviation
Baseline	40.05	44.95
ARIMA	25.53	19.68
SARIMA	21.51	21.74
Linear Regression	13.86	14.92
Support Vector Regression	14.14	16.80
Random Forest Regression	11.42	11.41
Neural Networks	16.36	17.68

Table 1. Long-term temporal prediction - Average and Standard Deviation of the Mean Absolute Percentage Error (MAPE) for each prediction model in the validation set

with an additional explicit daily seasonal component ($p = 7$ as the number of time bins in a day in our case).

Linear Regression. We fit a linear model, by finding the coefficients that multiplies each feature.

SVR. In our experiments, we use a Support Vector Regression (SVR) model with the following combination of parameters, which produced the best results among the values we tested: $C = 1000$, $\gamma = 0.1$, and $\epsilon = 0.1$, with the RBF kernel.

RFR. Random Forest Regression is an ensemble learning method that can be used for regression. The decision is based on the outcome of many decision trees, each of which is built with a random subset of the features. One advantage of random forests over linear regression is that the forest model is able to capture the non-linearity. Another advantage of RFR is that they are interpretable models, i.e. they offer a ranking of the most important features for the prediction problem. Here, we use 50 estimators (decision trees).

Neural Networks. We also consider a Long Short-Term Memory (LSTM) Neural Network model. LSTMs have a memory that may help capturing past trends in the data, which may favor our prediction task. We experimented with several different architectures. The best results were obtained with a three layer architecture where the input layer has 64 neurons (one for each feature), the dense layer has 4 neurons, and the output layer has one neuron. In our experiments, to balance prediction accuracy and training time, the model was trained for 50 epochs. Increasing the number of epochs has no significant impact on the accuracy of the model.

5.2. Long-term predictions - Results

Here we predict the FFCS demand for cars in the future months given a model built on the previous months. We use in our experiments the nine months of 2017 of car sharing usage of Vancouver. Given the volume of rentals in the training period, we try to predict the number of rentals in the validation period. For that, we use a model that is trained once and then used to perform all the predictions in the validation period. Our training set consists in the volume of rentals for six months in each bin of the day, and the validation data consists of volume of rentals for the next three months.

Table 1 shows the average mean absolute percentage error (MAPE) and the standard deviation of the MAPE for each of the prediction models. The models that rely only on the time series (ARIMA and SARIMA) are able to capture some patterns in the data, as their performance is considerably better than the baseline. However, the multivariate models perform better, with Random Forest Regression reaching the best performances. In Figure 7 we show the comparison between the actual values and the prediction in one month of the validation set using the Random Forest Regression model (orange dashed line). Overall the model is able to predict quite well the daily and weekly periodicity of rentals, but in general slightly underestimates the actual number of rentals. This could be due to the fact the training period refers to the first six months of the year, during which the average number of rentals is lower than during the validation period in fall.

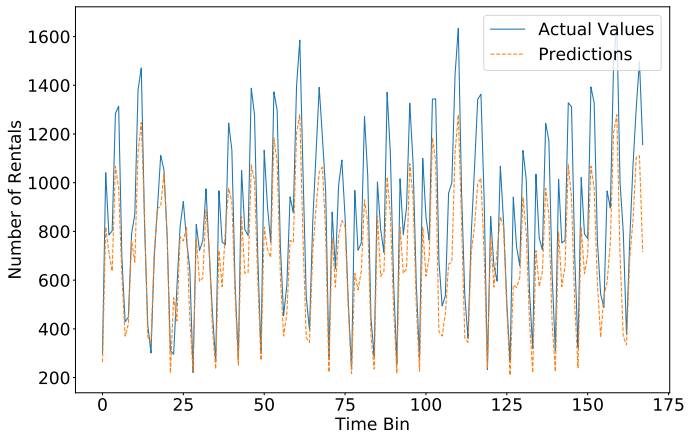


Figure 7. Long-term temporal prediction - Performance of the RFR model in one month of the validation set

	Expanding Window (starting: 28 days)		Sliding Window (28 days)	
	MAPE [%] Average	**MAPE [%] Standard Deviation**	**MAPE [%] Average**	**MAPE [%] Standard Deviation**
Baseline	20.12	16.64	20.12	16.64
ARIMA	36.01	35.87	36.52	36.60
SARIMA	17.60	20.01	18.02	21.75
Linear Regression	18.28	20.38	18.11	20.55
Support Vector Regression	12.22	15.62	12.87	18.52
Random Forests Regression	9.71	8.34	10.08	12.23
Neural Networks	10.52	12.93	10.52	12.74

Table 2. Short-term temporal prediction - Average and Standard Deviation of the Mean Absolute Percentage Error (MAPE) for each prediction model in the validation set

5.3. Short-term predictions - Results

We now tackle the problem of predicting the demand of cars in a city in the next time bin. Differently from the long-term predictions we use adaptive models, hence the model is re-trained every time new data is made available, so then we can add it to the training set. We here focus on the following prediction task: given the volume of rentals per time bin period for a specific number of past days and the weather conditions, we wish to predict the number of rentals just in the next time bin period.

We study this prediction task using two approaches: expanding window and sliding window. In the *expanding window* approach, after making the first prediction we add the actual value to the training set, therefore increasing the amount of data available for training in the next step. To train our models, we first set aside 21 days of data for validation, and start with 28 days of training data. In the *sliding window* approach, after making the prediction we remove the oldest training data and add the actual value to the training set. Therefore, the training set size is always the same during the evaluation of the models. To train our models, we consider different sliding windows sizes (from 7 to 28 days), and validate on the same validation set of 21 days as with the expanding window.

In Table 2, we compare the performance of all models using the two approaches. The best results for the sliding window approach were obtained with the largest possible window (28 days). The expanding window approach offers slightly better results, likely because the model can exploit more data and the patterns are not changing rapidly in time. Again, the multivariate models and in particular

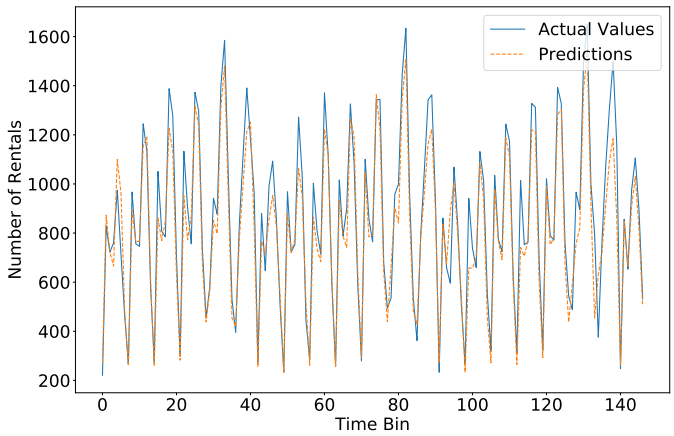


Figure 8. Short-term temporal prediction - Performance of the RFR model with expanding window in the validation set (21 days)

the Random Forest Regression models reach the best performance. Interestingly, the Neural Network model performs similarly to other models, suggesting that, for this specific use case, a simple and more interpretable model like a RFR is enough. We show in Figure 8 the performance of the best model, i.e., RFR with expanding window. In this short-term formulation of the problem the prediction naturally adapts to changes over time, obtaining better predictions with respect to long-term prediction. Moreover, the weather condition information also add useful information.

We now explore the importance of each feature for the model, by analyzing the RFR feature ranking. When training a tree, it can be computed how much each feature decreases the weighted impurity in a tree. For a forest, the impurity decrease from each feature can be averaged and the features can be ranked according to this measure. This gives a simple and interpretable feedback on which features are most useful for the prediction. We find that the most important features that impact the model are: (i) if we are in the daily peaks from 3 pm to 9 pm, (ii) during the night (0 am - 6 am) or (iii) if we are on a Friday and Saturday. Interestingly, the most important weather condition for the regressors is the presence of clouds, while the second one is a (rare) condition of presence of fog, mist and rain in the considered time bin.

6. Spatial prediction of rentals with socio-demographic data

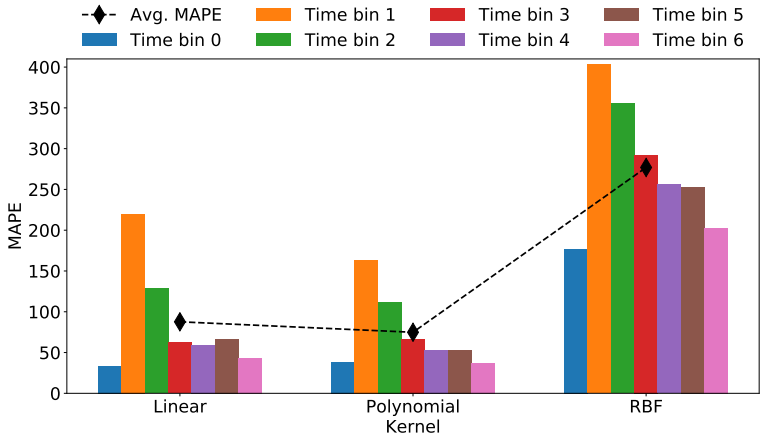
We now shift our attention to predict the demand of cars in a neighborhood without using past data as features. In other words, given only socio-demographic data in the neighborhoods, we want to predict the number of rentals that would start and end in others neighborhoods, and during each different period of the day. This problem is often referred to as a green field, or cold start, approach. In this case, no historical data is available, and the operator is interested in knowing which could be the possible system usage in a new neighborhood (or even a new city) based only on socio-demographic data.

Since we have very few data points for the training step (22 neighborhoods), we performed a leave-one-out validation. Given a target neighborhood, we consider information from all other neighborhoods for training the learning model, and consider the neighborhood that we left for validation.

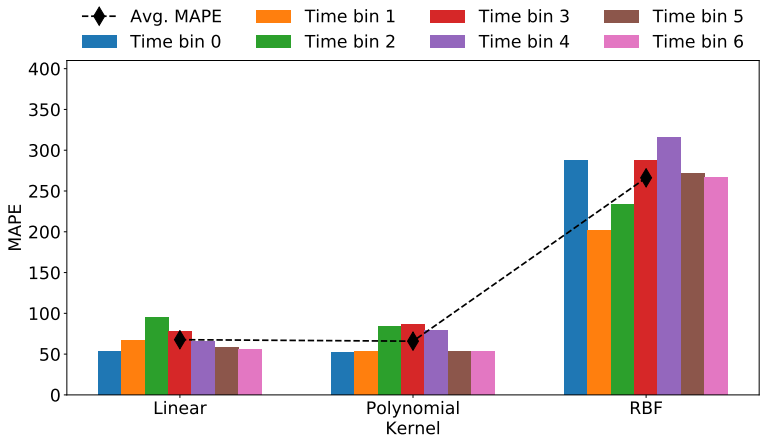
We manually selected 83 socio-demographic features that we think might be related to human mobility. Here, we only apply Support Vector Regression and Random Forest Regressions models, since here we are not dealing anymore with time series and RFR and SVR were the best performing in the temporal prediction. We discarded neural networks since these usually do not work well with a

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2019 doi:10.20944/preprints201911.0149.v1

Peer-reviewed version available at Electronics 2020, 9, 72; doi:10.3390/electronics9010072



(a) SVR Model: Starting rentals



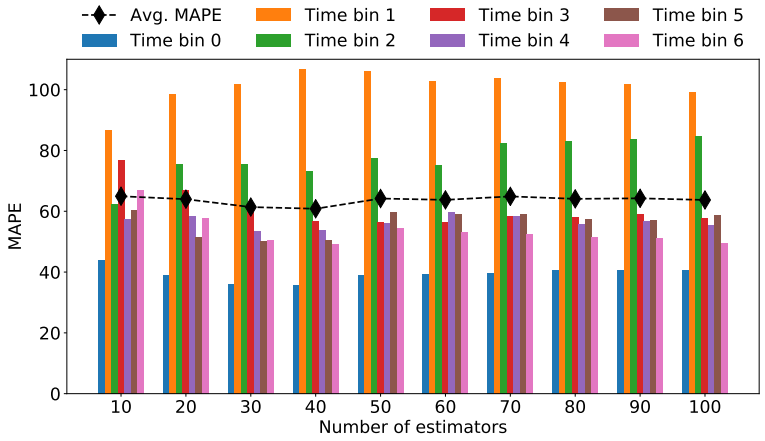
(b) SVR Model: Ending rentals

Figure 9. Spatial prediction - MAPE for Support Vector Regression models for different kernels

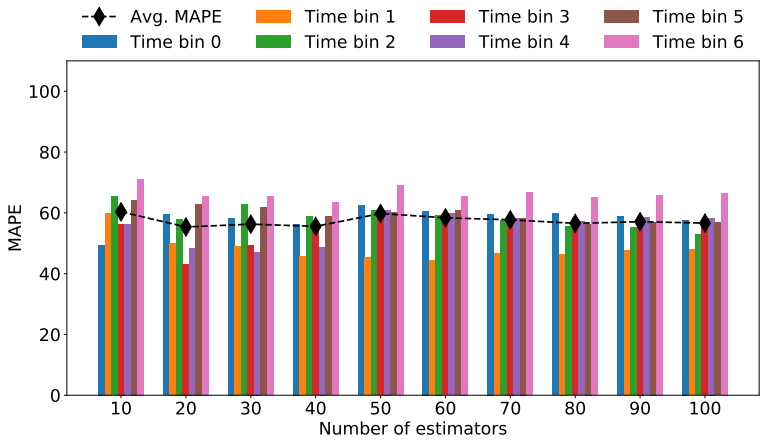
very small training set as in this case. For SVR, we tried 3 different kernels (linear, polynomial and RBF), with different combinations of parameters. The best performances were obtained for $\epsilon = 0.1$, $C = 100$ ($C = 10$ for RBF), and $\gamma = \frac{1}{\#features}$ ($\gamma = 1$ for RBF). For RFR, we tried number of estimators ranging from 10 to 100.

Figures 9a and 10b show the SVR prediction accuracy of the number of starting and ending rentals, respectively. For each kernel type, we report the average MAPE over the 22 neighborhoods for each time bin we considered in our analyses. SVR model performs poorly regardless the parameter setting we used or the time bin we target for prediction task. Best accuracy results occur with polynomial kernel, obtaining MAPE 70%, for starting rentals predictions and MAPE 64%, for ending rentals prediction. Time bin from 0 am to 6 am is the one with the best performances.

Random Forest Regression model results are shown in Figures 10a and 10b. For a given time bin and varying the number of trees MAPE does not vary that much, indicating that even 10 trees could be enough. Predictions for ending rentals perform better than those for starting rentals: the best case is the one with number of trees equals to 40 for predicting starting rentals, reaching MAPE 59% (20 trees for ending rentals, with MAPE 56%). Again, in the time bin from 0 am to 6 am we obtain the best predictions while the worst are obtained from 6 am to 9 am for starting rentals prediction. Differently from the starting rentals predictions, MAPE does not suffer from high fluctuation.



(a) RFR Model: Starting Rentals



(b) RFR Model: Ending Rentals

Figure 10. Spatial prediction - MAPE for Random Forests Regression models for different values of estimators

Rank	Feature	Relevance
1	Number of emergency calls	0.0717
2	Distance from downtown	0.0481
3	People commuting by walk	0.0381
4	People commuting within Vancouver	0.0342
5	People with income between 100 000 and 149 999 $CAD	0.0298
6	People with income between 60 000 and 69 999 $CAD	0.0286
7	People legally recognized as couple	0.0281
8	People with income more than 150 000 $CAD	0.0274
9	People divorced	0.0261
10	People commuting within the same neighborhood	0.0249
11	Couples having more than 3 children	0.0239
12	People with age between 50 and 54 years	0.0233
13	Unemployed people	0.0231
14	People never married	0.0217
15	People with income between 80 000 and 89 999 $CAD	0.0211

Table 3. Spatial prediction - Most relevant features and their importance for the prediction using Random Forest Regression. The first 7 are the ones that for obtain the best overall model

6.1. Feature ranking and selection

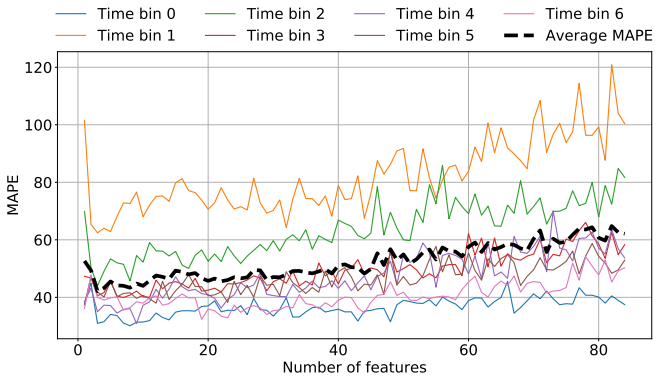
As in the previous section, we analyze the RFR ranking of the features. Table 3 reports the top-15 most relevant features. Hence, thanks to the RFR model, we hint at which data the operator should focus on when considering new neighborhood of the city to implement the FFCS. Such ranking allows us to reduce the number of features to train the model itself by selecting the most important ones. We run again the RFR with increasing number of features, chosen according to the given rank. We chose a-priori the number of trees, according to the best average MAPE obtained in Figure 10a and 10b. Thus, we choose 40 trees for the starting and 20 for the ending rentals prediction. Figure 11 shows the results. The horizontal axes represents the number of used features, while the vertical axes reports the MAPE. Notice the U-shaped curve of the average MAPE (dashed black line). Intuitively, too few features worsen the regression performances due to lack of information. Too many features also reduce the performance since the training is more complicated. In the following, we select the best number of features that minimize the average MAPE, which results to selecting the top 7 features in Table 1. With this subset, the average MAPE is 41% for starting rentals, 39% for arrivals.

At last, we explore the spatial prediction error, i.e., we look if there are neighborhoods that present significantly higher errors than others. Figure 12 depicts the heatmap of the MAPE per neighborhood, averaged over all time bins. The more the area is red the higher the MAPE is. Each green dot represents actual positions of starting or arrival rentals as recorded in the original trace. The areas having the highest error are the one labelled 15, 18, 11 and 0. We can see that most of these are periphery neighborhoods that only partially intersect with the rental area of the FFCS operator. This mismatch confuses the prediction since our model assumes the operative area coincides with the total area of each neighborhood. In a nutshell, our model predict much higher rentals (reflecting the whole neighborhood area) than the ones that are actually done (reflecting the restricted operational area). This offers the FFCS operator the opportunity to consider in which areas to extend the service.

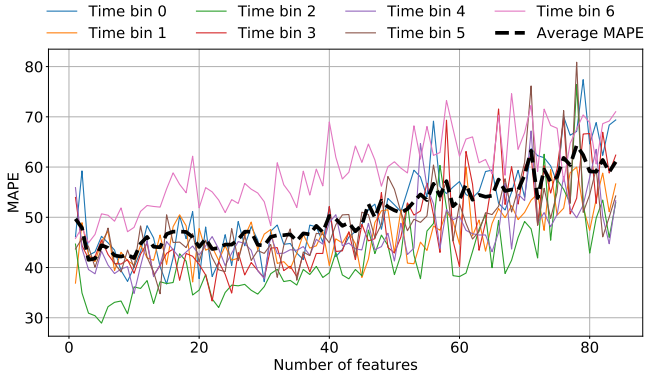
7. Conclusions

This paper studied the problem of predicting FFCS demand patterns, which is relevant to an adequate provisioning of the service. Relying on data from real FFCS rides in Vancouver as well as the municipality socio-demographic information, we investigated to which extent modern machine learning based solutions allow one to predict the transportation demand.

Our results showed that the temporal prediction of rentals can be solved with relative errors down to 10%. Here a simple Random Forests Regression performs consistently among the best models,



(a) Rentals starting



(b) Rentals arrivals

Figure 11. Spatial prediction - MAPE in the different time bins by selecting the most relevant features in RFR

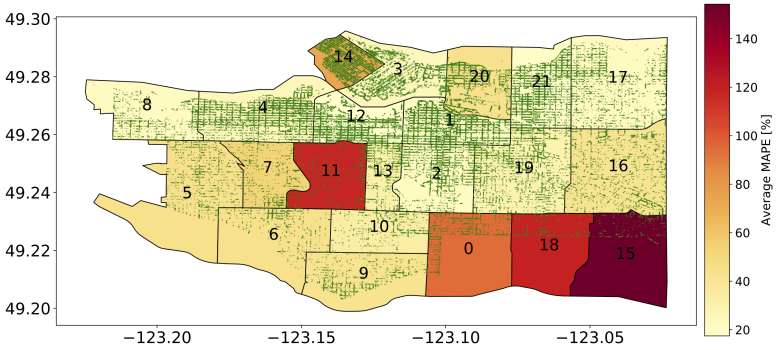


Figure 12. Spatial distribution - Heatmap of average MAPE per neighborhood. Rentals are shown on the map as green points

and allowing us also to discover which features are mostly affecting the prediction. When considering the mobility spatial prediction using socio-demographic data only, we obtain relative errors around 40%, after feature selection. Again, using a Random Forest Regression model, we can observe which features are the most useful for the prediction, a precious information for providers and regulators to understand FFCS systems and to provide a high-quality service that benefits both providers and its costumers.

References

1. Litman, T. Evaluating Carsharing Benefits. *Transportation Research Record* **2000**, *1702*, 31–35.
2. Firnkorn, J.; Müller, M. What will be the environmental effects of new free-floating car-sharing systems? The case of car2go in Ulm. *Ecological Economics* **2011**, *70*, 1519–1528.
3. Firnkorn, J.; Müller, M. Selling mobility instead of cars: new business strategies of automakers and the impact on private vehicle holding. *Business Strategy and the environment* **2012**, *21*, 264–280.
4. Ciari, F.; Weis, C.; Balac, M. Evaluating the influence of carsharing stations' location on potential membership: a Swiss case study. *EURO Journal on Transportation and Logistics* **2016**, *5*, 345–369.
5. Cocca, M.; Giordano, D.; Mellia, M.; Vassio, L. Free Floating Electric Car Sharing: A Data Driven Approach for System Design. *IEEE Transactions on Intelligent Transportation Systems* **2019**, pp. 1–13. doi:10.1109/TITS.2019.2932809.
6. Cocca, M.; Giordano, D.; Mellia, M.; Vassio, L. Free floating electric car sharing design: Data driven optimisation. *Pervasive and Mobile Computing* **2019**, *55*, 59 – 75. doi:10.1016/j.pmcj.2019.02.007.
7. Ciociola, A.; Cocca, M.; Giordano, D.; Mellia, M.; Morichetta, A.; Putina, A.; Salutari, F. UMAP: Urban Mobility Analysis Platform to Harvest Car Sharing Data. Proocedings of IEEE Smart City Innovations, 2017.
8. Brockwell, P.J.; Davis, R.A. *Introduction to time series and forecasting*; springer, 2016.
9. Bishop, C.M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*; Springer-Verlag: Berlin, Heidelberg, 2006.
10. Okutani, I.; Stephanedes, Y.J. Dynamic prediction of traffic volume through Kalman filtering theory. *Transportation Research Part B: Methodological* **1984**, *18*, 1–11.
11. Clark, S. Traffic prediction using multivariate nonparametric regression. *Journal of transportation engineering* **2003**, *129*, 161–168.
12. Ciari, F.; Schuessler, N.; Axhausen, K.W. Estimation of carsharing demand using an activity-based microsimulation approach: model discussion and some results. *International Journal of Sustainable Transportation* **2013**, *7*, 70–84.
13. Catalano, M.; Casto, B.L.; Migliore, M. Car sharing demand estimation and urban transport demand modelling using stated preference techniques. *European Transport* **2008**, *40*, 33–50.
14. Firnkorn, J.; Müller, M. Selling mobility instead of cars: new business strategies of automakers and the impact on private vehicle holding. *Business Strategy and the environment* **2012**, *21*, 264–280.
15. Becker, H.; Ciari, F.; Axhausen, K.W. Comparing car-sharing schemes in Switzerland: User groups and usage patterns. *Transportation Research Part A: Policy and Practice* **2017**, *97*, 17–29.
16. Schmöller, S.; Bogenberger, K. Analyzing external factors on the spatial and temporal demand of car sharing systems. *Procedia-Social and Behavioral Sciences* **2014**, *111*, 8–17.
17. Schmöller, S.; Weikl, S.; Müller, J.; Bogenberger, K. Empirical analysis of free-floating carsharing usage: The Munich and Berlin case. *Transportation Research Part C: Emerging Technologies* **2015**, *56*, 34–51.
18. 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* **2019**, p. 101448. doi:https://doi.org/10.1016/j.is.2019.101448.
19. Jain, R. *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*; John Wiley & Sons, 1990.
20. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, E. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **2011**, *12*, 2825–2830.