Predicting Bikeshare System Usage Up to One Day Ahead

Romain Giot, Raphael Cherrier

Predicting Bikeshare System Usage Up to One Day Ahead

Romain Giot
Univ. Bordeaux, LaBRI, UMR 5800, F-33400 Talence, France
CNRS, LaBRI, UMR 5800, F-33400 Talence, France

Raphaël Cherrier
QUCIT, F-33400 Talence, France
www.qucit.com

Abstract—**Bike sharing systems are present in several modern cities. They provide citizens with an alternative and ecological mode of transportation, allowing them to avoid the use of personal car and all the problems associated with it in big cities (*i.e.*, traffic jam, roads reserved for public transport, . . .). However, they also suffer from other problems due to their success: some stations can be full or empty (*i.e.*, impossibility to drop off or take a bike). Thus, to predict the use of such system can be interesting for the user in order to help him/her to plan his/her use of the system and to reduce the probability of suffering of the previously presented issues. This paper presents an analysis of various regressors from the state of the art on an existing public dataset acquired during two years in order to predict the global use of a bike sharing system. The prediction is done for the next twenty-four hours at a frequency of one hour. Results show that even if most regressors are sensitive to over-fitting, the best performing one clearly beats the baselines.**

I. Introduction

Bike sharing systems are present in more than 600 cities around the world and allow citizens to rent a bike in one station and drop it off in another one in order to travel across the town [1]. This new way of travelling is getting more and more popular as many cities open their own schemes. It is also an efficient way of displacement as Jensen *et al.* [2] have shown (at least for the scheme of Lyon) that its use is most of the time faster than the use of a personal car. Although the economical model differs among the implementations, most schemes share very similar technologies, and it is interesting to be able to predict the future usage of the system. Indeed, thanks to a prediction system, the user would be able to better plan his/her trips. This way he/she is almost sure to walk to a station only if there are free bikes and to release the bike in a station where there are free slots.

Such system would also be useful for the administrators of the bike sharing system, to know in advance when it will be necessary to manually exchange the bikes between two stations when one station is full whereas the other one is empty. This way the administrator is able to manage the system to prevent situations where a user cannot bring back his/her bike at a full station or cannot take a bike at an empty station. In both cases the user experience is improved and the system is more pleasant to use and will be more adopted.

In this work, we want to provide a system able to predict the number of bikes used (or inversely the number of free slots) some hours in advance. The problem can be stated as a regression problem which takes in input various features and produces the number of bikes in use for the requested delay. The information used corresponds to features extracted from various sources (weather, previous usage of the bike sharing system, calendar) available openly in any town (*i.e.*, information not proprietary, except the usage of the bike sharing system). The proposed system is validated on a dataset acquired on the bike sharing system of Washington DC [3], but it could be reproduced for another town as it does not need any information specific to Washington DC.

The proposed system is compared to several baseline methods already used in the literature which have been designed in order to perform well and to be usable without the need of a computer. We show that the proposed methods perform better than these baselines, and thus would be better than a guess from a human operator. However, we also show that the problem is really sensitive to over-fitting and this point must be taken seriously.

Although other studies about the prediction of bike sharing systems have already been done, this paper improves the state of knowledge because:

- the proposal is evaluated on a two years dataset [3] whereas most studies of the literature validated their results only on datasets collected during few weeks;
- we analyse the use of machine learning tools whereas most studies of the literature use signal processing tools. As their theoretical backgrounds are different, these approaches are complementary;
- we have observed the prediction for different deltas between the current moment and the hour for which we want to predict the usage ; most of the time this delta is larger than in any study;

The paper is organised as follows. Section II presents the previous works related to the use of computational intelligence systems for bike sharing systems. Section III presents the experimental protocol as well as the proposed method. Section IV presents the obtained results. Section V discusses these results. Section VI concludes the paper.

II. Computational Intelligence for Bike Sharing Systems

Several works have been done on bike sharing systems, but not all the studies are related to the prediction of the use of the system in the future. They can be related to the balancing of the bikes or the characterization of the use of the system.

In most bike sharing systems, it is necessary to manually balance the bikes among the stations in order to avoid full or empty stations. Thus, at a different frequency, trucks take bikes from some stations and distribute them to other ones. This step is named the "balancing of the stations". Several researchers proposed algorithms to reduce the cost of such operation. Benchimol *et al.* [4] provides an algorithm in $O(logC + nlogN)$ with N the number of stations, n the number of bikes and C the number of trucks. Chemla *et al.* propose an algorithm where the balancing is operated by only one truck [5].

Fanaee-T and Gama [3] use anomaly detection to detect events thanks to the bike sharing usage information, weather and working day information. The system has been applied to Washington DC on a dataset acquired on a period of 2 years (it is the dataset we use in our work). Their system is able to detect 30 events which have a real meaning. They have also made some predictions on the use of the bike sharing system but do not provide information in order to reproduce the experiment (*i.e.*, train and validation splitting, parameters, ...).

Yoon *et al.*[6] propose a personal journey advisor for navigating in Dublin using the bike sharing system. One of its key module consists in predicting the availability of bikes in stations in a near future with a spatio temporal prediction system based on AutoRegressive Integrated Moving Average (ARIMA) which also takes into account seasonal trends and spatial correlations. The system has been evaluated on a dataset of 3 weeks.

Froehlich *et al.* [7] also provide a system to predict the number of bikes available in a near future (from 10 to 120 minutes) with a Bayesian Network. The method has been evaluated on a dataset acquired on a period of 13 weeks.

Borgnat *et al.* [8] build a model of the cyclic temporal patterns with a linear regression and used it to do forecasting. Variables come from weather, number of users, holidays markers. The database is the biggest one: more than two years.

Kaltenbrunner *et al.* [9] use a predictor based on Auto Regressive Moving Average which takes into account the availability of bikes in the near stations. The evaluation is done on a dataset of 7 weeks.

Table I summarizes the main studies in bike sharing prediction. Please note that the lack of standardised evaluation procedure (data, duration, error metric, ...) forbids to do a fair comparison between them.

III. Experimental Protocol

A. Use of a Public Dataset

We use a public dataset provided by Fanaee-T and Gama [3]. They have produced it, using another one released by Capital Bikeshare[1], by computing the number of bikes in use every hour from individual trips made during the years 2011 and 2012 (17379 hours). In opposite to other studies, they do not provide the information station per station, but for the system in a whole. So, in opposite to most of the other works, the prediction have a city granularity instead of a station granularity. However, it is the sole public dataset which proposes both information about the bike sharing system and the environmental conditions and we could expect that our results would be reproducible when applied to datasets representing individual stations.

In addition to the number of bikes used per hour, the dataset provides various additional information: the season (spring, summer, fall, winter), if it is banked holiday in the US, the weekday, if it is a working day (not a weekend and not a holiday day), the type of weather (among 4 categories), the temperature, the feeling temperature, the humidity, the wind speed, and the number of bikes used by casual users, registered users and both of them. The maximum number of bikes used is 977.

B. Data Cleaning and Features Extraction

The number of bikes used in the system is a timeseries which values must be correlated with the previous ones or at least partially depend on them. For this reason, we need to use this chronological information if we want to build a good prediction system. Thus, we cannot directly work with the raw dataset presented before, but we need to build time dependant features. Among those presented in section III-A, we are not interested in the number of casual or registered users in this study. The number of bikes in use is the information we want to predict, while the other features serve to do this prediction.

There are some holes within the provided dataset (*i.e.* there is not one sample for all the hours of the acquisition period). This first operation consists in adding artificial samples in order to ensure that we have really one sample per hour for each day (we added 165 samples among the 17544 to obtain this clean dataset[2]). The missing features are duplicated from the value of the previous hour (season, weather, ...) or computed from the date of the day (weekday, ...). Note that these artificial samples serve only for the next step, but are never used in the training or validation process. However, this modification may have a negative impact on the recognition performance as the extracted features (explanation follow) are computed with inaccurate information.

Now that we have a clean dataset with one sample per hour, we add additional features for each samples:

- the week number in the calendar;
- the number of bikes that were available one hour before; ($Bikes_{1h\ ago}$), two hours before ($Bikes_{2h\ ago}$), ... up to twenty-four hours before ($Bikes_{24h\ ago}$) the time for which we will make a prediction.

With these features, we will attempt to predict the number of bikes in use up to twenty-four hours ahead.

The features used to do the prediction is summarized in Table II, and each of them is represented by a real value. We need the meteorological information of the moment that we want to compute the prediction. As this moment is in the

[1]http://www.capitalbikeshare.com/trip-history-data

[2]it is less than 1%

Table I
SUMMARY OF EXISTING WORK ON USAGE PREDICTION AND PRESENTATION OF BEST RESULT

Authors	City	Timespan	Methods	Error metric
Fanaee-T and Gama [3]	Washington DC	2 years	Weka's regressors	Relative Absolute Error (29.98%), Root Relative Squared Error (39.27%)
Yoon *et al.*[6]	Dublin	27 days	modified ARIMA	Root Mean Square Error. 5 min: 0.91, 60 min: 3.47
Froehlich *et al.* [7]	Barcelona	13 weeks	Bayesian network	Relative Absolute Error:0.08
Borgnat *et al.* [8]	Lyon	2 years + 8 months	Linear Regression	Mean Relative Error: 12%
Kaltenbrunner *et al.* [9]	Barcelona	7 weeks	Auto-Regressive Moving Average	Mean Absolute Error: 1.39

future when we make the prediction, we make the assumption that we use a weather forecast service which performs well up to 24h ahead (its error is small compared to other sources of error in our prediction).

C. Baseline Regression Systems

The aim of an intelligent system is to take an ensemble of features and to predict the number of bikes in used in fixed delay (with one regressor per delay). It is thus a regression problem for which several algorithms are available in the literature. We have chosen to use three baseline classifiers to compare to the regression systems:

- *Baseline Mean Value*, the mean number of bikes on the training dataset;
- *Baseline Mean Hour*, the mean number of bikes on the training set at this specific hour (similar to Historic Mean in [7]).
- *Baseline Last Hour*, the number of bikes of the latest hour available (similar to Last Value in [7]);

These baselines are quite fair as they use simple rules that seem logical and could be manually done by operators. Our objective is to use and configure automatic algorithms which beat them.

D. Selected Regression Systems

Several regression algorithms exist in the literature. We have selected several of them to find which one could do the best prediction. Please refer to their original paper for deeper implementation details. Each regressor takes as input $\vec{x}$ the feature vector of interest and produces a number which is a prediction of the number of bikes in use. The tested regressors are:

- *Ridge Regression* which corresponds to a Linear Regression with a $L2$ regularization term. This is aimed at reducing over-fitting by favoring simpler models with lower regression coefficients. The regression function is $f(\vec{x}) = \sum_i w_i \vec{x}_i + w_0$ and the weights $\vec{w}$ are computed by optimising: $\mathrm{argmin}_{\vec{w}} ||X \cdot w - y||_2^2 + \alpha \cdot ||w||_2^2$ with X the training samples. α controls the power of the regularization.
- *Adaboost Regression* [10], [11] which uses a boosting of several random weak decision tree regressors: a first weak regressor is fitted on the original training dataset, while other successive regressors are fitted by iteratively increasing the weights of the samples which do not perform well with the previous weak regressors. This way the latests weak regressors focus on the most difficult samples.
- *Support Vector Regression (SVR)* [12] on standardized features (removal of the mean and scaling to unit variance) which searches within the training set a subset of samples named the support vectors in order to compute the regression with them, with the kernel trick. The training step searches the parameters α_i, α_i^*, b and the n support vectors $x_i, 0 < i <= n$ in order to build the regression function: $f(\vec{x}) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) kernel(\vec{x_i}, \vec{x}) + b$.
- *Random Forest Regression* [13] in which a set of random regression trees are built on different bootstraps of the training set. The regression value is the mean of the output of all the regression trees.
- *Gradient Tree Boosting Regression* [14] where the training incrementally builds the regression function by progressively adding to it a new decision tree which minimizes the best the error metric. The generated regression function is $f(\vec{x}) = \sum_{m=1}^{M} \gamma_m h_m(\vec{x})$ with h_m the m-*th* weak learner.

Note that none of these regressors have been tested in the previous works [3], [6], [7], [8], [9]. Table III presents the list of parameters tested for each of them.

Figure 1 is an overview of the proposed system. Yellow corresponds to the feature extraction and dataset splitting. Blue corresponds to the training of the regressors. Pink corresponds to the predicting. Green corresponds to the evaluation.

E. Evaluation Procedure

In order to evaluate the performance of each prediction system, it is necessary to use two datasets: a training one and a validation one. However, we must ensure that the validation dataset only contains samples acquired after all the samples of the training dataset. For this reason, we have chosen to use a split date S and use all the samples acquired before as training samples and all the samples acquired after as validation samples. We have chosen to set S to split at $2/3$ of the dataset (2012-05-02 08:00:00) which gives 11 571 training samples and 5 808 validating samples.

Among the various metrics of the literature to evaluate regression systems, we have chosen to use the Root Mean Squared Difference (RMSD) error rate which is a standard

Table II
LIST OF FEATURES USED DEPENDING ON THE DELTA BETWEEN THE CURRENT TIME AND THE MOMENT WHEN WE WANT A PREDICTION. THE HIGHER THE DELAY, THE LOWER THE NUMBER OF FEATURES AVAILABLE

Delta	**Features**																																				
	HOLIDAY	SEASON	WORKING DAY	WEATHER	TEMP	ATEMP	HUMIDTY	WINDSPEED	WEEKDAY	MONTH	HOUR	WEEK	DAY	$Bikes_{1h\ ago}$	$Bikes_{2h\ ago}$	$Bikes_{3h\ ago}$	$Bikes_{4h\ ago}$	$Bikes_{5h\ ago}$	$Bikes_{6h\ ago}$	$Bikes_{7h\ ago}$	$Bikes_{8h\ ago}$	$Bikes_{9h\ ago}$	$Bikes_{10h\ ago}$	$Bikes_{11h\ ago}$	$Bikes_{12h\ ago}$	$Bikes_{13h\ ago}$	$Bikes_{14h\ ago}$	$Bikes_{15h\ ago}$	$Bikes_{16h\ ago}$	$Bikes_{17h\ ago}$	$Bikes_{18h\ ago}$	$Bikes_{19h\ ago}$	$Bikes_{20h\ ago}$	$Bikes_{21h\ ago}$	$Bikes_{22h\ ago}$	$Bikes_{23h\ ago}$	$Bikes_{24h\ ago}$
1H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
2H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
3H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
4H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓				✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
5H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓					✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
6H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓						✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
7H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
8H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
9H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓									✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
10H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓										✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
11H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓											✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
12H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓												✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
13H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓													✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
14H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓														✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
15H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓															✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
16H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓																✓	✓	✓	✓	✓	✓	✓	✓	✓
17H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓																	✓	✓	✓	✓	✓	✓	✓	✓
18H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓																		✓	✓	✓	✓	✓	✓	✓
19H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓																			✓	✓	✓	✓	✓	✓
20H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓																				✓	✓	✓	✓	✓
21H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓																					✓	✓	✓	✓
22H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓																						✓	✓	✓
23H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓																							✓	✓
24H	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓																								✓

Table IV
PERFORMANCE (LOWER IS BETTER) OF EACH PREDICTION SYSTEM FOR THE 24 DELAYS. THE BEST PERFORMING SYSTEM FOR EACH DELAY IS PRESENTED IN BOLD. "+" MARKS SYSTEMS PERFORMING BETTER THAN THE BEST BASELINE, "-" MARKS SYSTEMS PERFORMING WORST THAN THE BEST BASELINE, "=" MARKS THE BEST BASELINE.

Delay	Baseline Mean Hour	Baseline Last Hour	Baseline Mean Value	Adaboost Regressor	Ridge Re-gression	SVR	Random Forest Regressor	Gradient Boosting Regressor –
1	182.87^{-}	129.82$^{=}$	243.11^{-}	102.46^{+}	**79.32**$^{+}$	336.78^{-}	336.17^{-}	312.69^{-}
2	182.87$^{=}$	210.22^{-}	243.11^{-}	119.68^{+}	**111.10**$^{+}$	253.13^{-}	336.18^{-}	319.47^{-}
3	182.87$^{=}$	255.19^{-}	243.11^{-}	130.95^{+}	**120.06**$^{+}$	252.05^{-}	335.46^{-}	336.71^{-}
4	182.87$^{=}$	282.21^{-}	243.11^{-}	132.14^{+}	**122.19**$^{+}$	252.95^{-}	334.24^{-}	339.08^{-}
5	182.87$^{=}$	305.58^{-}	243.11^{-}	130.51^{+}	**122.82**$^{+}$	255.89^{-}	333.89^{-}	296.52^{-}
6	182.87$^{=}$	328.80^{-}	243.11^{-}	132.64^{+}	**122.89**$^{+}$	256.06^{-}	334.69^{-}	302.97^{-}
7	182.87$^{=}$	345.35^{-}	243.11^{-}	130.92^{+}	**122.78**$^{+}$	301.80^{-}	334.16^{-}	283.62^{-}
8	182.87$^{=}$	348.27^{-}	243.11^{-}	131.42^{+}	**122.71**$^{+}$	257.63^{-}	334.01^{-}	303.35^{-}
9	182.87$^{=}$	339.47^{-}	243.11^{-}	127.18^{+}	**123.47**$^{+}$	259.53^{-}	334.14^{-}	298.51^{-}
10	182.87$^{=}$	338.63^{-}	243.11^{-}	130.16^{+}	**123.55**$^{+}$	264.22^{-}	333.92^{-}	320.67^{-}
11	182.87$^{=}$	347.19^{-}	243.11^{-}	126.55^{+}	**123.55**$^{+}$	267.86^{-}	333.46^{-}	273.94^{-}
12	182.87$^{=}$	352.33^{-}	243.11^{-}	128.42^{+}	**123.65**$^{+}$	331.55^{-}	333.72^{-}	307.68^{-}
13	182.87$^{=}$	348.52^{-}	243.11^{-}	131.92^{+}	**123.59**$^{+}$	267.14^{-}	333.50^{-}	334.65^{-}
14	182.87$^{=}$	340.52^{-}	243.11^{-}	130.24^{+}	**123.48**$^{+}$	266.43^{-}	333.58^{-}	285.30^{-}
15	182.87$^{=}$	339.72^{-}	243.11^{-}	130.03^{+}	**123.44**$^{+}$	266.37^{-}	332.98^{-}	321.72^{-}
16	182.87$^{=}$	345.24^{-}	243.11^{-}	127.88^{+}	**123.41**$^{+}$	266.05^{-}	332.96^{-}	254.92^{-}
17	182.87$^{=}$	340.58^{-}	243.11^{-}	131.11^{+}	**123.42**$^{+}$	269.20^{-}	330.49^{-}	285.88^{-}
18	182.87$^{=}$	324.18^{-}	243.11^{-}	129.86^{+}	**123.43**$^{+}$	271.12^{-}	325.19^{-}	332.40^{-}
19	182.87$^{=}$	302.62^{-}	243.11^{-}	128.64^{+}	**123.45**$^{+}$	269.39^{-}	324.63^{-}	317.59^{-}
20	182.87$^{=}$	282.13^{-}	243.11^{-}	129.86^{+}	**123.51**$^{+}$	268.83^{-}	323.91^{-}	317.66^{-}
21	182.87$^{=}$	260.09^{-}	243.11^{-}	131.61^{+}	**123.62**$^{+}$	269.02^{-}	323.98^{-}	329.64^{-}
22	182.87$^{=}$	226.07^{-}	243.11^{-}	130.84^{+}	**123.71**$^{+}$	338.86^{-}	323.81^{-}	321.40^{-}
23	182.87^{-}	173.61$^{=}$	243.11^{-}	134.16^{+}	**123.71**$^{+}$	338.51^{-}	322.65^{-}	321.40^{-}
24	182.87^{-}	134.17$^{=}$	243.11^{-}	129.61^{+}	**123.88**$^{+}$	336.61^{-}	325.48^{-}	291.67^{-}

Table III
PARAMETERS TESTED FOR EACH REGRESSOR DURING THE GRID SEARCH STEP. ONLY THE BEST PERFORMING SET OF PARAMETERS IS USED ON THE VALIDATION DATASET.

Regressor	**Parameters**
Ridge Regression	• α: 0.01, 0.1, 1, 10, 100
Adaboost Regression	• Number of estimators: 10, 50, 100, 400 • Loss function: linear, square, exponential
Support Vector Regression (SVR)	• Regularisation parameter (C): 1, 10, 100, 1000 • Kernel: RBF • γ : 1e-3, 1e-4, 1/#nb features
Random Forest Regression	• Number of estimators: 10, 50, 100, 400 • Maximum number of features: $\sqrt{\text{\# nb features}}, log2(\text{\# nb features})$
Gradient Tree Boosting Regression	• Number of estimators: 10, 50, 100, 400 • Maximum number of features: $\sqrt{\text{\# nb features}}, log2(\text{\# nb features})$ • Learning rate: 0.5, 0.75, 1

error rate for regression problems and is coherent for our problem:

$$RMSD = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - a_i)^2} \quad (1)$$

with n the number of scores to predict, p_i the prediction and a_i the ground truth of sample i.

As the selected regression algorithms need various parameters which are not trivial to configure, we use a grid search approach with cross validation in order to automatically select the best parameters. For each regressor, we have built a list of configuration parameters (Table III). Thanks to a 3 folds cross validation scheme, each configuration parameter of each regressor is tested and scored with the RMSD on the training dataset (thus the error rate is computed on samples which have not been used to train the model with which we compute the prediction). For each regressor, we keep the best performing configuration (the one which minimizes the RMSD) and retrain the whole training dataset with it. Then, the prediction is made on the validation dataset with each regressor configured with its best parameters. The evaluation is also done with the RMSD.

IV. RESULTS

Table IV presents the performance of each prediction system for the 24 different delays (between the moment we make the prediction and the moment we want to predict) and Figure 2 allows to compare the performance on the training and validation datasets.

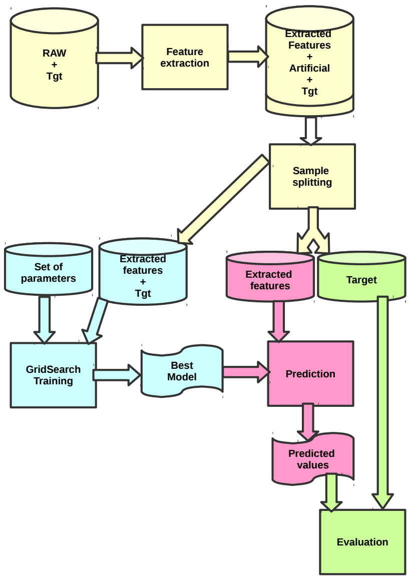
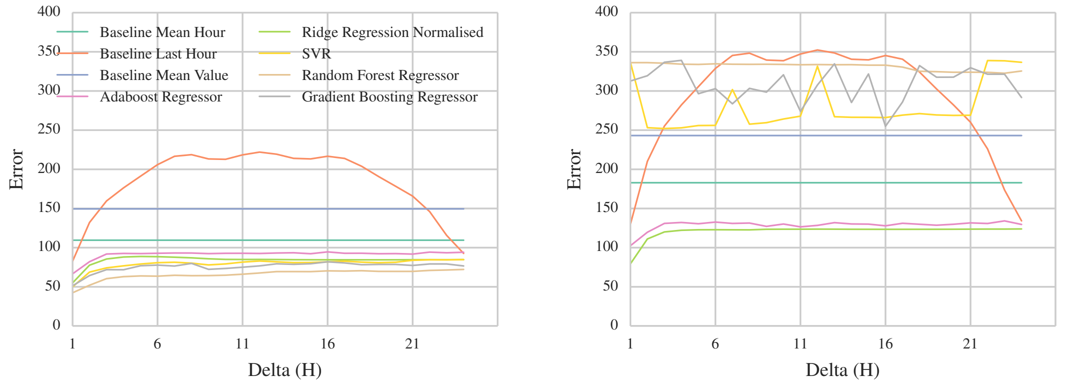


Figure 1. Overview of the whole system. Yellow area corresponds to feature extraction, blue area to training, pink area to testing and green area to evaluating

The first three columns correspond to the baselines. "Baseline Mean Value" is never the best baseline while "Baseline Last Hour" is the best baseline when the delay is of 1 hour (due to short term correlations) or 23 and 24 hours (due to a partial periodicity in the phenomenon under study), but it is the worst in most other cases. For all other delays, the best baseline is "Baseline Mean Hour" with a constant error rate of 182.87. A regression system is considered to be good only if it provides a prediction better than the best baseline for all delays.

The next columns correspond to the intelligent systems. The best performing baseline performs always worse than the two best performing regressors ("Ridge Regression" and "Adaboost regressor") while "SVR", "Random Forest Regressor" and "Gradient Boosting Regressor" always perform worse than the best baseline. From Figure 2 we see that the bad performance of the bad regressors clearly come from overfitting. Indeed, they perform very well in the training dataset (Figure 2(a)) whereas they are the worst on the validation dataset (Figure 2(b)). The superiority of the best regressors is supported by a statistical analysis thanks to the Wilcoxon signed-rank test. "Ridge Regression" is also statistically better than "Adaboost Regressor". They both perform better when the delay is of 1 hour; errors of "Ridge Regression" tend to increase with the delay, but no trend is observed for "Adaboost Regressor".

The number of bikes depends on the hour of the day, although there are few correlations between the hour and the number of bikes in use (0.39 with Pearson correlation coefficient). This can be shown with the low error rate of "Baseline Mean Hour" and the shape of the "Baseline Last



(a) Training error (b) Validation errors

Figure 2. Evolution of the error rate depending on the prediction delay on the training and validation datasets. We can observe over-fitting for several regressors.

Hour" where the error rate is lower when the delta is small or close to one day (we can assume that we would obtain something similar on a longer range of delays of test each 24 hours). The "Baseline Mean Hour" error value is higher in the validation dataset than in the training dataset. It shows that we cannot really make assumptions of previous mean bike usage to predict the future bike usage, because too many factors have to be taken into account.

Figure 3 presents the difference between the groundtruth and each prediction system for different delays on a subset of moments to predict. The best prediction systems are the systems closer to 0. The peaks (*i.e.* the big variations between the true and the predicted value) tend to increase when the delta grows and tend to be decrease when the delta come close to 24 hours. However we cannot notice understandable patterns.

Figure 4 presents the prediction of the two best systems and the groundtruth for several delays. There is a very close match between the prediction and the groundtruth with a delay of 1 hour. The differences tend to be larger when the delay is different of 1 but seem to be stable (*i.e.*, it is not worst when the delay increases) over time (this is also confirmed by Figure 2).

V. Discussion

Whereas most studies of the literature use datasets of several stations with a sampling frequency of few minutes, we used a dataset with global information on the city and a frequency of one hour. For this reason it will be necessary to validate the experiment on other datasets in order to generalize these results and show we can predict the individual use of each station.

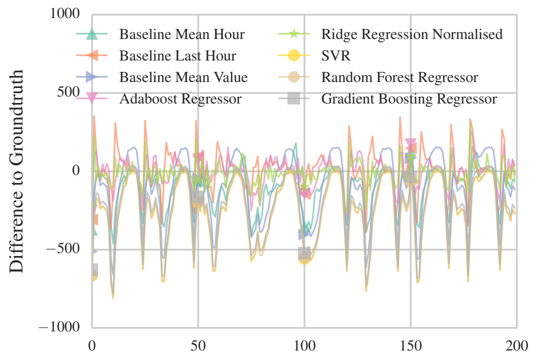
It would be necessary to investigate deeper on the reason why several machine learning tools suffer from over-fitting. Indeed, we have used several state of the art algorithms which are supposed to be resistant to over-fitting and most of them performed badly because of over-fitting. The protocol should be changed in order to avoid the over-fitting issue by modifying the kind of extracted features computed or the parameters of the regressors. It is important to tackle this problem as better performances can be obtained with such regressors.

It would also be interesting to use more regressors which try to reduce the feature space in order to analyse which features are the more pertinent and use only them with the other regressors in order to increase the prediction performance. Lasso with its $l1$-norm could be a good candidate. Even if the "Ridge Regressor" does not try to eliminate features, the analysis of the weight it computes can be interesting. By analysing the weights of the "Ridge Regressor" applied on normalised samples (so each features as a similar importance), for the delay of one hour, we can sort the features by their relevance:

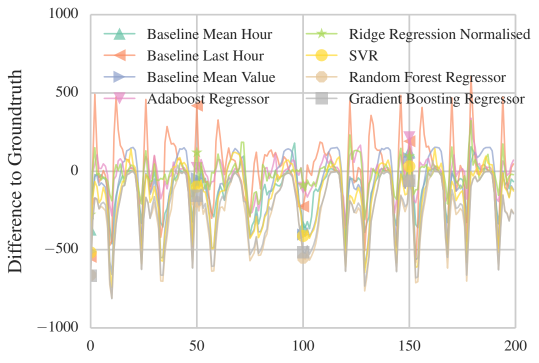
- Very relevant ($weight > 100$): $Bikes_{1h\ ago}$,
- Relevant ($100 \geq weight > 10$): $Bikes_{2h\ ago}$, $Bikes_{9h\ ago}$, $Bikes_{16h\ ago}$, $Bikes_{17h\ ago}$, $Bikes_{8h\ ago}$, $Bikes_{23h\ ago}$, $Bikes_{10h\ ago}$, $Bikes_{24h\ ago}$, $Bikes_{18h\ ago}$, $Bikes_{15h\ ago}$, TEMP,
- Average ($10 \geq weight > 0$): $Bikes_{13h\ ago}$, $Bikes_{5h\ ago}$, $Bikes_{3h\ ago}$, WEATHER, $Bikes_{12h\ ago}$, MONTH, $Bikes_{11h\ ago}$, $Bikes_{21h\ ago}$, $Bikes_{4h\ ago}$, $Bikes_{7h\ ago}$, $Bikes_{14h\ ago}$, $Bikes_{20h\ ago}$, $Bikes_{22h\ ago}$, HUMIDTY, $Bikes_{6h\ ago}$, ATEMP, SEASON, WEEK, $Bikes_{19h\ ago}$, WEEKDAY, DAY, HOLIDAY,
- Not relevant ($weight \approx 0$): HOUR, WINDSPEED, WORKING DAY,

However, we have to be careful in our interpretation of these results. Indeed, we used a linear regressor which means a feature will show up as relevant in this classification only if it is linearly relevant. For example, the HOUR variable is classified as "not relevant" not because it is truly irrelevant but because a linear classifier fails to grasp its influence as it is highly non-linear.

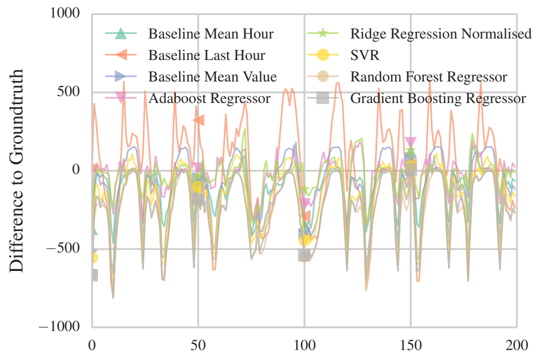
It would also be interesting to see if removing the low weighted features from the experiment would increase the



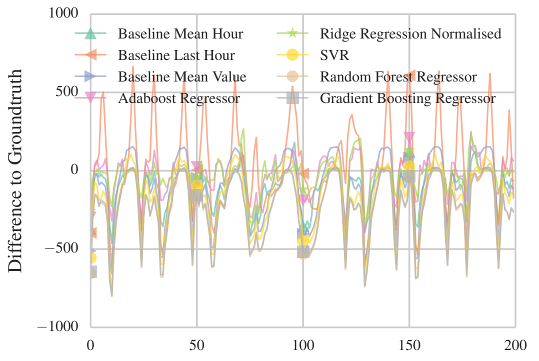
(a) 1H



(b) 2H



(c) 20H



(d) 24H

Figure 3. Difference between the prediction of each algorithm and the ground truth on the first 200 hours of the validation dataset

performance of the other regressors.

From Table II we can see that the number of features used to make the prediction decreases when the delay increases. It would be interesting to use a deeper memory for these cases in order to always have 24 different hours of features whatever is the delay. The use of a bigger memory could improve the recognition performance. Note, that this improvement is not guaranteed because the actual difference between the delays of 1 and 2 hours is very large.

The weather information of the moment for which we want a prediction is necessary (remember that the temperature has a big impact). It would be interesting to verify if this information is really useful or of it could be possible to use a slightly outdated value (*i.e.* a value really acquired before and not predicted by a weather prediction system).

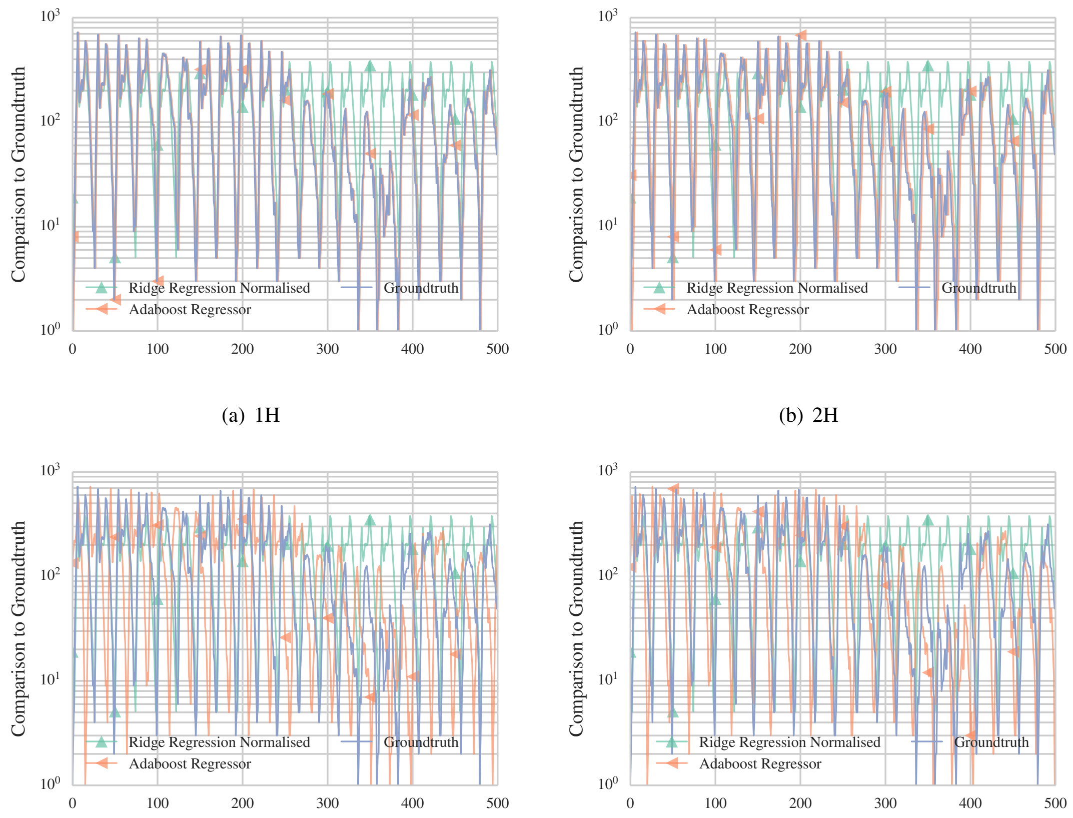
Despite of the issues and possible improvements to do, we think that using the best regressor we could provide a good idea of the future use of the bike sharing system to the user and help him/her to choose his/her way of transportation (bike if there is a low usage or something else otherwise).

VI. Conclusion

In this paper, we have shown on a real world dataset that it is possible to predict the use of bike sharing system up to 24 hours ahead. Various regression systems have been tested and have shown performances better than intuitive baseline systems which could be manually used by bike sharing system administrators. The features used are different when the prediction must be done in one hour or later and is based on the weather information, previous bike usage information and holiday information.

The proposal have been evaluated on a public dataset which contains 2 years of information on the Washington DC bikeshare system. The two best performing systems are based on the use of Ridge Regression and Adaboost Regression. They both always outperform the best baseline. Although these methods works quite well, the error ratio is higher when the delay is superior to one hour. We have shown there is an important issue of over-fitting with other regressors of the state of the art.

In a near future, we want to verify if we can draw the same conclusion on a larger number of datasets and if it is possible to add additional interesting information in order to improve the performance. It is also necessary to work on a station granularity instead of a town granularity in order to help the maintainers of the bike sharing system to balance their bikes. The use of incremental regressor must be investigated has it could reduce the over-fitting problems.



(a) 1H (b) 2H

(c) 20H (d) 24H

Figure 4. Comparison between the prediction of some of the best methods and the ground truth on the last 500 hours of the validation dataset (notice the logarithmic scale)

Acknowledgments

We want to thank the author of the dataset we use within this paper [3] as well as the authors of the library scikit-learn [15].

References

[1] P. DeMaio, "Bike-sharing: History, impacts, models of provision, and future," *Journal of Public Transportation*, vol. 12, no. 4, pp. 41–56, 2009.

[2] P. Jensen, J.-B. Rouquier, N. Ovtracht, and C. Robardet, "Characterizing the speed and paths of shared bicycle use in lyon," *Transportation research part D: transport and environment*, vol. 15, no. 8, pp. 522–524, 2010.

[3] H. Fanaee-T and J. Gama, "Event labeling combining ensemble detectors and background knowledge," *Progress in Artificial Intelligence*, pp. 1–15, 2013.

[4] M. Benchimol, P. Benchimol, B. Chappert, A. De La Taille, F. Laroche, F. Meunier, L. Robinet *et al.*, "Balancing the stations of a self-service bike hire system," *RAIRO-Operations Research*, vol. 45, no. 1, pp. 37–61, 2011.

[5] D. Chemla, F. Meunier, and R. Wolfler Calvo, "Bike sharing systems: Solving the static rebalancing problem," *Discrete Optimization*, vol. 10, no. 2, pp. 120–146, 2013.

[6] J. W. Yoon, F. Pinelli, and F. Calabrese, "Cityride: a predictive bike sharing journey advisor," in *Mobile Data Management (MDM), 2012 IEEE 13th International Conference on*, 2012, pp. 306–311.

[7] J. Froehlich, J. Neumann, and N. Oliver, "Sensing and predicting the pulse of the city through shared bicycling." in *IJCAI*, 2009, pp. 1420–1426.

[8] P. Borgnat, P. Abry, P. Flandrin, C. Robardet, J.-B. Rouquier, and E. Fleury, "Shared bicycles in a city: A signal processing and data analysis perspective," *Advances in Complex Systems*, vol. 14, no. 03, pp. 415–438, 2011.

[9] A. Kaltenbrunner, R. Meza, J. Grivolla, J. Codina, and R. Banchs, "Urban cycles and mobility patterns: Exploring and predicting trends in a bicycle-based public transport system," *Pervasive and Mobile Computing*, vol. 6, no. 4, pp. 455–466, 2010.

[10] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[11] H. Drucker, "Improving regressors using boosting techniques," in *ICML*, vol. 97, 1997, pp. 107–115.

[12] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," *Advances in neural information processing systems*, vol. 9, pp. 155–161, 1997.

[13] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[14] J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, 2002.

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