Nearest-Neighbor Distance Method Applied to Model Reuse With Bike Rental Station Data

Fernando Simeone, Diego S. Mendes and Ahmed A. A. Esmin

Department of Computer Science, UFLA, MG, Brazil
fernandosimeone@posgrad.ufla.br, diego.mendes@posgrad.ufla.br,
ahmed@dcc.ufla.br

Abstract. We describe our submission to the ECML/PKDD 2015 Model Reuse with Bike Rental Station Data Discovery Challenge, where the objective is to predict the number of available bikes in every bike rental stations 3 hours in advance and the main task is to reuse the models learned on 200 "old" stations in order to improve prediction performance on the 75 "new" stations. It exploits a selective approach, together with automatic reuse model selection from the old station models.

1 Introduction

Regression techniques are widely used in scenarios such as the retail market, stock exchange, decision support systems, among others. However, it is still a big challenge to create techniques that perform good predictions fully automatically, requiring often prior knowledge of the data to improve the accuracy of such applications.

Thus, an obstacle commonly encountered in the application of regression techniques is the difference between the collected data, which are used for training a predictive model to the real scenario in which such models are applied, resulting prediction errors. This is a scenario that seen recurring, in many cases, because it is not always able to collect data from the real context to construct a specific model for it. This makes it important to study strategies for re-use regression models in different scenarios, adapting their use as needed in search of more assertive predictions.

In this context it is in this work, which aims to provide a solution to the challenge posed by the ECML / PKDD 2015 called " MoReBikeS: Model Reuse with Bike Rental Station data"[1]. The challenge is to predict, with three hours in advance, the number of bicycles available for bike rental stations. The main challenge of the problem lies in finding the best way to re-use regression models built using historical data of another set of stations.

The work is divided as follows: the Section 2 explains briefly the problem posed by the challenge of ECML / PKDD 2015; the Section 3 presents the strategies proposed here to solve this problem; the Section 4 presents the results obtained and the Section 5 presents the conclusions. and future work.

[1] http://reframe-d2k.org/Main_Page

2 The Problem

The challenge " MoReBikeS: Model Reuse with Bike Rental Station data ", which was presented as one of the challenges of the ECML / PKDD 2015, is to provide with 3 hours in advance the number of bicycles available in a bicycle rental station. This prediction has real practical applications, one of them for the benefit of the rental company, since predicting that stations will be completely filled or emptied, the company can relocate the bikes to better serve users. Another application benefits the user, who can know in advance in which station it can make the lease or return of bicycles, noting the likely number of bikes or vacancies, respectively.

In the scenario presented in the challenge there are 275 stations, being 200 old stations $(1, 2, \ldots, 200)$, for which were collected various data every hour over a period of two years, and 75 other new stations $(201, 202, \ldots, 275)$, which were only collected data from one month. Considering the static characteristics, geographical position and the number of docks of each station was available. For the time series data collected every hour, there are data related to time (date, time, day of week, holiday, etc.), weather (wind speed and direction, temperature, humidity, etc.) and bicycles available in the station (number of bikes, bike number 3 hours, bicycle average to date, etc.).

For each of the 200 old stations, based on historical data, it was built and made available 6 linear regression models, which consider different attributes to predict the number of available bikes. Table 1 displays the models available for each station(columns) and attributes of the stations used by them (lines).

Table 1. Features and their use in each of the models.

Features	short	short_temp	full	full_temp	short_full	short_full_temp
bikes_3h_ago	✓	✓	✓	✓	✓	✓
short_profile_3h_diff_bikes	✓	✓			✓	✓
short_profile_bikes	✓	✓			✓	✓
temperature.C		✓		✓		✓
full_profile_3h_diff_bikes			✓	✓	✓	✓
full_profile_bikes			✓		✓	✓

The features shown in Table 1 are included among the data of the historical series of the stations. In other words, a record of the series contains all those features. To understand, here is a brief description of some of them:

- `bikes`: number of present bikes at the station at the time (this is the attribute to be predicted for the new stations);
- `bikes_3h_ago`: number of present bikes at the station three hours ago;
- `full_profile_bikes`: arithmetic mean of the attribute `bikes` of the entire history of the station, at the same time of the week;
- `full_profile_3h_diff_bikes`: arithmetic average value calculated `bikes` − `bikes_3h_ago` in the entire history of the station, at the same time of the week;
- `short_profile_bikes` and `short_profile_3h_diff_bikes`: Similar to the `full_profile_bikes` and `full_profile_3h_diff_bikes` attributes, but consider only the last 4 records with the same time of the week, and not the entire history;

The challenge is to find the best way to select, among this total of 1200 models, which will be used and how to combine them to predict how many bikes there in each of the 75 new stations, 3 hours in advance.

3 Proposed Strategies

The focus of this work was to find a simple and effective strategy to predict the number of bicycles of the new stations using the pre-existing models of others. To achieve this goal, it considered the following premise:

Bike stations have similar characteristics with others that are geographically close.

Thus, from a previous knowledge of the proposed issue, it was noted that nearby stations have similar characteristics, such as climate, topography and culture, which can be crucial to find out the number of users using the rental service bicycles [1].

Therefore, it was decided to use a dissimilarity metric widely used in classification and clustering techniques: the Euclidean distance. When using this metric can be considered any attributes of the objects to be compared, but here it was decided to use only the geographical attributes of the station (latitude and longitude). Thus, the distance (dissimilarity) between two stations is given by Formula 1.

$$EuclideanDist(Station_A, Station_B) = \sqrt{(lat_A - lat_B)^2 + (long_A - long_B)^2} \tag{1}$$

Moreover, it was not considered that there would be an ideal model among the existing ones, would perform good predictions for all new stations. Therefore, the strategies used in this study sought to find, within a set of pre-existing models, which would be most suitable to the new station. It was considered that the most appropriate models for a given station would be those of the closest stations to her, obtained using Formula 1.

Therefore, two strategies were used to predict the number of bicycles of new stations, which will be described below.

3.1 Strategy 1: Use models of nearest neighbor station

The main idea of this strategy is to seek more similar station (with the smallest distance) the new station. Having possession of that station, two different approaches were used, which are described below.

The first approach is to use a single pre-defined model for predictions. Considering that M is the set of models available for each station, should be informed which model M_i to be used. After finding the nearest station, the model M_i of this station will be used for the predictions of the new station S_{new}. The procedure of this approach is described in Algorithm 1. In this and the following, the variable *instance* is the station record of the S_{new} you want to predict the number of bicycles.

Algorithm 1 Closest Station Single Model Strategy

```
1: function ClosestStationSingleModel(S_new, M_i, instance)
2:     closestStation ← findClosestStation(S_new)
3:     model ← closestStation.getModel(M_i)
4:     prediction ← model.predict(instance)
5:     prediction ← AdjustPrediction(prediction, S_new)
6:     return prediction
7: end function
```

As you may notice a prediction adjustment technique is performed as indicated in the Algorithm 2, to avoid being provided more bikes that fit in the new station, as well as negative predictions. Such adjustment strategy was used in all strategies and approaches used in this work.

Algorithm 2 Adjust Predictions

```
1: function AdjustPrediction(prediction, S_new)
2:     if prediction > S_new.maxDocks then
3:         prediction ← S_new.maxDocks
4:     else if prediction < 0 then
5:         prediction ← 0
6:     end if
7:     return prediction
8: end function
```

The second approach is to, rather than to use only one model, consider all models available from the nearest station. Thus, the prediction of the new station S_{new} consists in the aggregation of the predictions for each model of the nearest station. Three forms of aggregation of the results were used: average, median and average no outliers. This approach is described in Algorithm 3.

In the Algorithm 3 must be pre-established the form of aggregation of the results (mean, median or average no outlliers). Thus, the algorithm proceeds as

Algorithm 3 Closest Station Multiple Models Strategy

```
1: function ClosestStationMultiModels(S_new, aggrStrategy, instance)
2:     closestStation ← findClosestStation(S_new)
3:     predictions ← [ ]
4:     models ← closestStation.getAvailableModels()
5:     for model in models do
6:         prediction ← model.predict(instance)
7:         prediction ← AdjustPrediction(prediction, S_new)
8:         predictions[model] ← prediction
9:     end for
10:    return predictions.aggregateResults(aggrStrategy)
11: end function
```

in the previous approach, but finding the next station, stores the predictions of each model in a vector (*predictions*). After this step, such predictions are combined using the informed aggregation strategy (*aggrStrategy*).

3.2 Strategy 2: Use of models from neighboring stations within a radius

The idea of this strategy assumes that it may be more advantageous to consider more than a neighborhood element, instead of using a single neighboring station. Thus, given a new station S_{new}, the stations whose distance from S_{new} is smaller than a radius r are obtained. The models obtained from these stations are used to make the predictions and the results are aggregated using a pre-established strategy (average, median or average no outliers). From this view, two approaches were used, as described below.

The first approach is to use all models of the selected stations. This strategy is described in Algorithm 4, where the results of the predictions of the selected models are stored in a matrix *predictions*, so that it can be aggregated. It can be seen that, if no station is found within r, the Algorithm 3 is applied, so the models of the nearest station are used. In this case, the predictions are aggregated using the strategy "average no outliers", with which the best results were obtained for this situation.

Another approach to this strategy is to use only a pre-established model of each selected station as shown in Algorithm 5. As can be noted, the only difference of approach used in Algorithm 5 is that only the models of type m_i are considered for each station within the radius r to make the predictions.

The Figure 1 shows, briefly, the idea of the second strategy, which was used in this work.

As can be noted, for this strategy, in both approaches, it is necessary to inform the radius to be used by the algorithm. This parameter is crucial for the effectiveness of the strategy. Therefore, to define the radius was made an exhaustive search in the range $[0.001, 0.01]$, which was defined based on running tests of the algorithm using the available training data.

Algorithm 4 Closest Stations at Radius Multiple Models Strategy

```
1: function StationsAtRadiusMultiModels(S_new, r, aggrStrategy, instance)
2:     closestStations ← findStationsAtRadius(S_new, r)
3:     if closestStations.length > 0 then
4:         for closeStation in closestStations do
5:             models ← closeStation.getAvailableModels()
6:             for model in models do
7:                 prediction ← model.predict(instance)
8:                 prediction ← AdjustPrediction(prediction, S_new)
9:                 predictions[closeStation.id, model] ← prediction
10:            end for
11:        end for
12:        return predictions.aggregateResults(aggrStrategy)
13:    else
14:        return ClosestStationMultiModels(
                       S_new, AVG_WITHOUT_EXTREMES, instance)
15:    end if
16: end function
```

Algorithm 5 Closest Stations at Radius Single Model Strategy

```
1: function StationsAtRadiusSingleModel(S_new, r, aggrStrategy, M_i, instance)
2:     closestStations ← findStationsAtRadius(S_new, r)
3:     if closestStations.length > 0 then
4:         for closeStation in closestStations do
5:             model ← closestStation.getModel(M_i)
6:             prediction ← model.predict(instance)
7:             prediction ← AdjustPrediction(prediction, S_new)
8:             predictions[closeStation.id] ← prediction
9:         end for
10:        return predictions.aggregateResusts(aggrStrategy)
11:    else
12:        return ClosestStationMultiModels(
                       S_new, AVG_WITHOUT_EXTREMES, instance)
13:    end if
14: end function
```

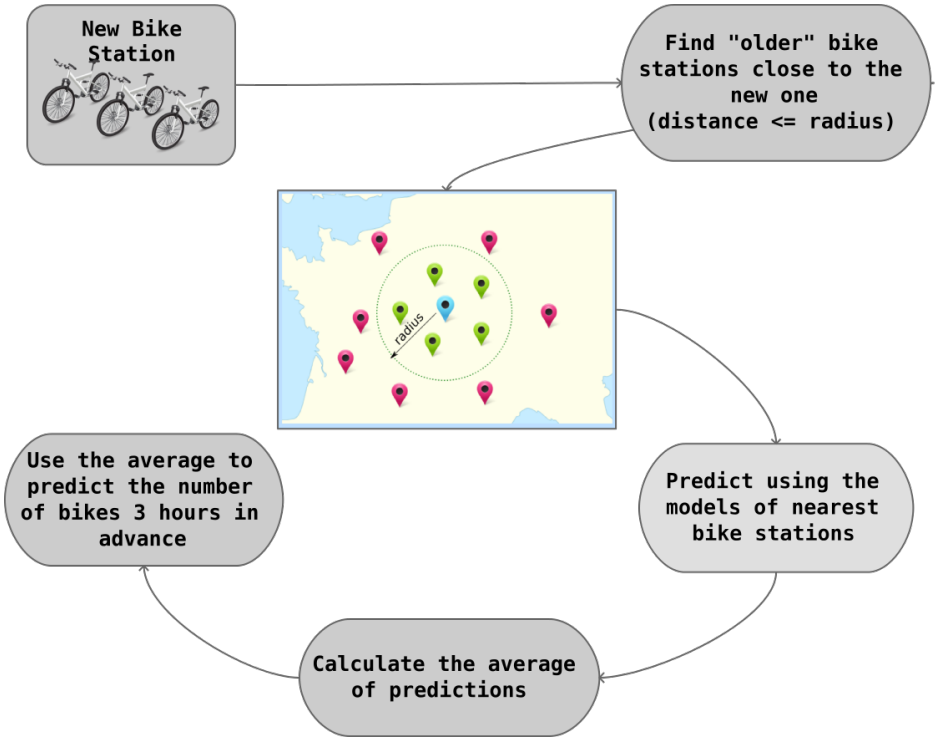

Fig. 1. Closest Stations at Radius with Multiple Models Strategy.

4 Results

In the evaluation of the proposed strategies, the historical series of 1 month available for the 75 new stations was used as test dataset. The data consists in 745 records per station, totalizing 55875 records. Thus, predictions were made to such records and we calculated the mean absolute error (MAE), as the Formula 2. Considering n records to be predicted, $predicted_i$ represents the i-th predicted value and $expected_i$ its expected value.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|predicted_i - expected_i| \tag{2}$$

In tests, the two proposed strategies were used, using each of the presented linear regression models separately, and also using all models together aggregating their values (using aggregation strategies mentioned above). The table 2 summarizes the errors obtained using each approach. It can be seen that the strategy 2 achieved better results in all configurations. The model "`short_full`" was the one that obtained the best results for both strategies.

For the implementation of the strategy 2, as described above, we need the r parameter, which specifies the maximum distance between stations whose mod-

Table 2. Obtained results in each strategy using each of the models.

Estratégia	**Estratégia 1**	**Estratégia 2** r=0.0061
`short`	2.8329	2.8087
`full`	2.8123	2.7719
`short_full`	**2.8049**	**2.7694**
`short_temp`	2.8885	2.8762
`full_temp`	2.8696	2.8397
`short_full_temp`	2.8450	2.8170
`All(Average)`	2.8247	2.7998
`All(Median)`	2.8248	2.7986
`All(Average without extremes)`	2.8241	2.7978

els will be used. The value used was $r = 0.0061$, which was obtained through exhaustive testing. The image 2 shows the mean absolute error values according to different values of r that were tested, taking in account that the tests were done considering all models.

The best configuration obtained by experiments was the use of Strategy 2, with the model "`short_full`", using the parameter $r = 0.0061$. This was the setting whose results were submitted to challenge "MoReBikeS: Model Reuse with Bike Rental Station data" in the ECML/PKDD 2015. Note that the results presented here differ from those released by the challenge organization because the test suite used for official submission did not contain actual results, and his error could be calculated only by the challenge organization.

5 Conclusions

In this work we present two solution strategies for the challenge "MoReBikeS: Model Reuse with Bike Rental Station data". Both started from the premise that nearby stations may have similar behaviors in regard to bicycle rental.

The good results of Strategy 2 shows that it is possible to make predictions reusing models built from the historical data of other stations, and that similar stations may have a similar behavior. The good performance of the model "`short_full`" shows that profiles calculated from historical data can be a good indicator for the predictions.

The strength of the presented strategies is that they can be applied in other contexts, as will seek to use the most appropriate models according to the scenario presented and their characteristics. Also, this is a simple strategy, easy implementation and low computational cost.

References

1. Arya, S., Mount, D.M., Netanyahu, N.S., Silverman, R., Wu, A.Y.: An optimal algorithm for approximate nearest neighbor searching fixed dimensions. Journal of the ACM (JACM) 45(6), 891–923 (1998)

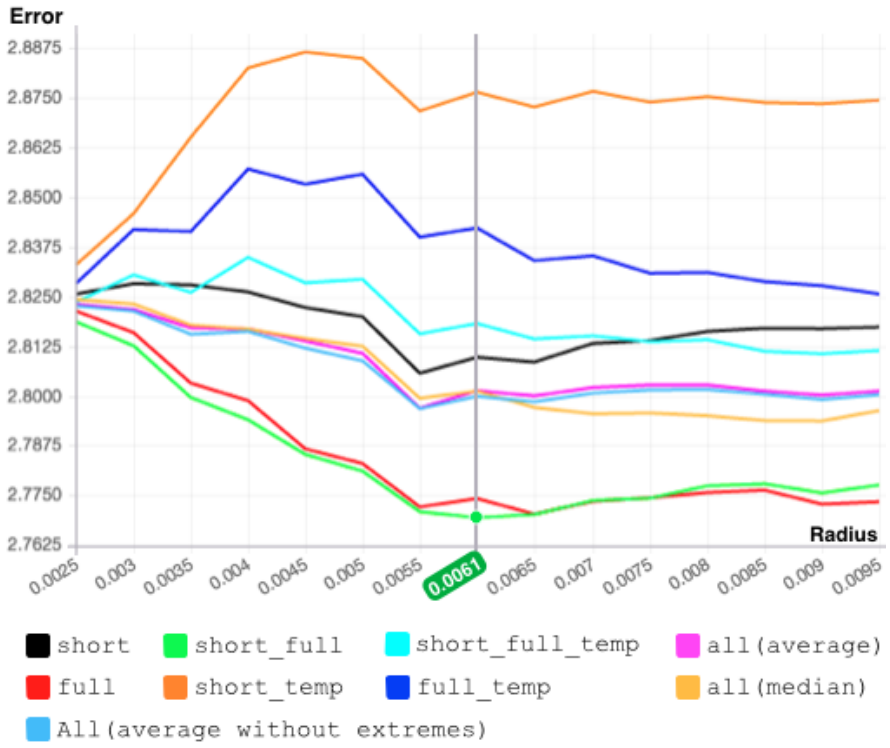

Fig. 2. Mean absolute error obtained according to the radius used in the Strategy 2.