More Bikes: Experiments in Univariate Regression

Tim Lawson

December 22, 2023

1 Task description

The assignment is to predict the number of available bikes at 75 rental stations in three hours' time for a period of three months beginning in November 2014, i.e., a supervised univariate regression problem. It is divided into three sub-tasks, which differ in the information that is available:

1. The number of available bikes at each of the 75 stations for the month of October 2014. This sub-task may be approached by building a separate model for each station or a single model for all 75 stations.
2. A set of linear models that were trained on the number of available bikes at each of a separate set of 200 stations for a year. For the first ten stations, this data is available for analysis but not training.
3. Both of the above.

The predictions are evaluated by the mean absolute error (MAE) between the predicted and true numbers of available bikes over the period of three months, beginning in November 2014. The evaluation data is not available to participants; however, the score achieved on a held-out test set is reported on the task leaderboard[1]. This report begins with a preliminary analysis of the data in section 2. Then, I describe the general approach taken to the sub-tasks in section 3. Finally, section 4 presents the results of the experiments for each sub-task.

2 Data analysis

The data is at hourly intervals with $n = 54385$ instances across 75 stations. A summary of its features and the distributional characteristics of the non-temporal quantitative features are given in tables 1 and 2 respectively. This shows that many of the features, including the target variable `bikes`, are missing for $n = 73$ instances, which were excluded from the analysis. Additionally, the 'profile' features, i.e., the features derived from the numbers of available bikes at preceding times, are not defined for the first week of instances at each station. Hence, they are missing for approximately $\frac{1}{4}$ of the instances. The meteorological features are constant for all stations at a given timestamp. Naturally, the number of available bikes at a given station is bounded by zero and the number of docks at that station.

2.1 Feature selection

Intuitively, features that have zero variance are not informative for regression analysis and were automatically excluded[2]. In the case of sub-task 1, the available data is limited to the month of October 2014; therefore, these included the month and year. Additionally, the 'station' features (table 1) are

[1] `https://www.kaggle.com/competitions/morebikes2023/leaderboard`

[2] See `sklearn.feature_selection.VarianceThreshold`.

Category	Feature	Data type	Kind
Station	station	int	ordinal
	latitude	float	
	longitude	float	
	docks	int	
Temporal	timestamp	int	
	year	int	
	month	int	
	day	int	
	hour	int	
	weekday	str	ordinal
	weekhour	int	
	is_holiday	bool	categorical
Meteorological	wind_speed_max	float	
	wind_speed_avg	float	
	wind_direction	float	
	temperature	float	
	humidity	float	
	pressure	float	
	precipitation	float	
Bikes	bikes	int	
	bikes_avg_full	float	
	bikes_avg_short	float	
	bikes_3h	int	
	bikes_3h_diff_avg_full	float	
	bikes_3h_diff_avg_short	float	

Table 1: A summary of the features. Except where indicated, the features are quantitative.

Feature	n/a	Mean	Variance	Skewness	Kurtosis
wind_speed_avg	73	4.69	21	1.26	1.59
wind_direction	365	170.23	7,553.8	$2.64 \cdot 10^{-2}$	−0.29
temperature	73	21.71	10.7	0.75	0.97
humidity	73	65.94	279.7	−0.57	−0.49
pressure	73	1,002.26	1,808.27	−2.53	5.55
bikes	73	7.35	43.15	1.1	1.3
bikes_avg_full	12,264	7.31	35.82	1.19	1.88
bikes_avg_short	12,264	7.31	35.82	1.19	1.88
bikes_3h	292	7.34	43.17	1.11	1.31
bikes_3h_diff_avg_full	12,483	$4.1 \cdot 10^{-3}$	22.4	0.25	10.54
bikes_3h_diff_avg_short	12,483	$4.1 \cdot 10^{-3}$	22.4	0.25	10.54

Table 2: The distributional characteristics of the non-temporal quantitative features. Features that have zero variance for the first case of sub-task 1 are excluded (section 2.1).

Figure 1: The Pearson correlation coefficients between pairs of quantitative features. The ordering of the features follows table 1. Features that have zero variance for the first case of sub-task 1 are excluded (section 2.1). The `timestamp` feature is also excluded because it is naturally correlated with the other temporal features.

constant for all instances at a given station; hence, for the first case of sub-task 1, these were excluded. Finally, the `precipitation` feature is zero for all instances.

Correlations between features and, more generally, multicollinearity, are undesirable in regression analysis (Alin 2010). Therefore, I sought to identify and exclude redundant features. To determine these, I computed the Pearson correlation coefficients between pairs of quantitative features (fig. 1). This analysis yielded the following observations:

- `bikes_avg_full` and `bikes_avg_short` are fully correlated ($r = 1.00$);
- `bikes_3h_diff_avg_full` and `bikes_3h_diff_avg_short` are fully correlated ($r = 1.00$); and
- `wind_speed_max` and `wind_speed_avg` are highly correlated ($r = 0.96$).

Hence, the second of each of these pairs of features was also excluded.

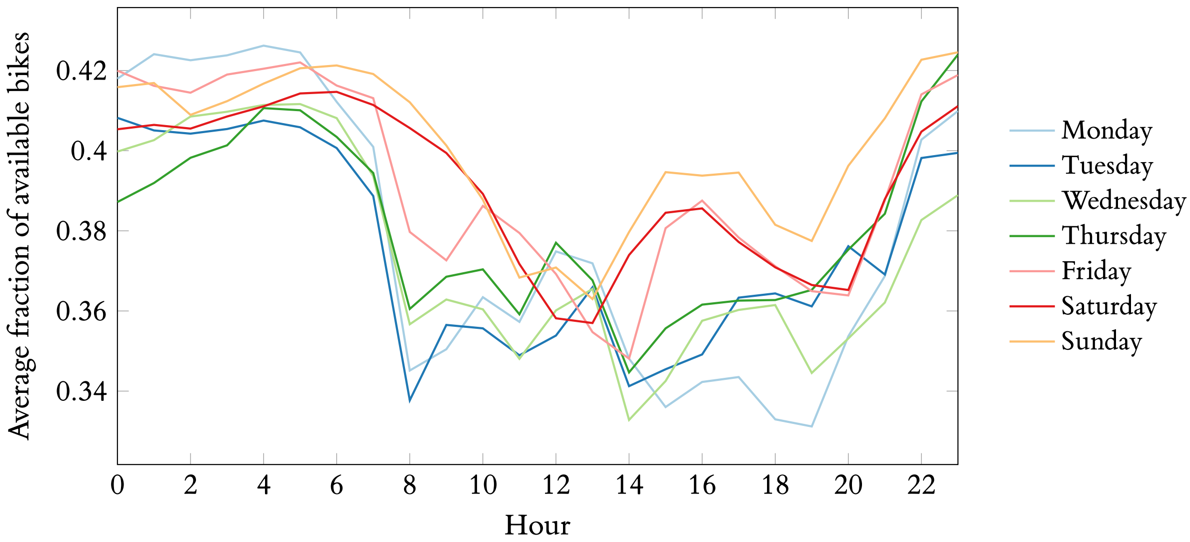

Figure 2: The average fraction of available bikes at each hour of the day, separated by the day of the week. This chart is limited to the month of October 2014.

2.2 Feature engineering

3 Methods

The experiments described in this report were performed with the `scikit-learn` Python package (Pedregosa et al. 2011). In each case, preprocessing and feature selection were performed by 'estimators' that implemented the 'transformer' interface; prediction was performed by estimators that implemented the 'predictor' interface; and estimators were composed into `Pipeline` objects over which hyperparameter search was performed (Buitinck et al. 2013, pp. 4–9). The bounds on the number of available bikes at a given station were enforced by predicting the *fraction* of bikes, i.e., the number of bikes divided by the number of docks at the station. In each case, this was implemented by extending the `TransformedTargetRegressor` meta-estimator to permit data-dependent transforms[3].

Generally, standard k-fold cross-validation is disfavoured for time-series data due to the inherent correlation between successive folds (Bergmeir et al. 2018). Instead, nested time-series cross-validation[4] with ten folds was performed to evaluate the models. This behaviour is illustrated in fig. 3. Hyperparameter tuning was performed by grid search[5]. The evaluation metric used throughout was the mean absolute error between the predicted and true numbers of available bikes, following the task description. Finally, the statistical significance of the differences in performance between models was assessed by paired t-tests of the scores achieved on each of the ten cross-validation folds.

Gradient-boosted decision trees are a popular choice for time-series forecasting problems (Bojer and Meldgaard 2021). In particular, the `HistGradientBoostingRegressor` estimator[6], inspired by the `LightGBM` implementation of gradient-boosted decision trees (Ke et al. 2017), supports missing values, which are evident in the data (table 2). Hence, for sub-task 1, I elected to focus on this class of models and to compare their performance to individual decision trees. As a baseline, I predicted the arithmetic mean fraction of available bikes at each station. The results of these experiments are presented in section 4.1.

[3] See `sklearn.compose.TransformedTargetRegressor`.
[4] See `sklearn.model_selection.TimeSeriesSplit`.
[5] See `sklearn.model_selection.GridSearchCV`.
[6] See `sklearn.ensemble.HistGradientBoostingRegressor`.

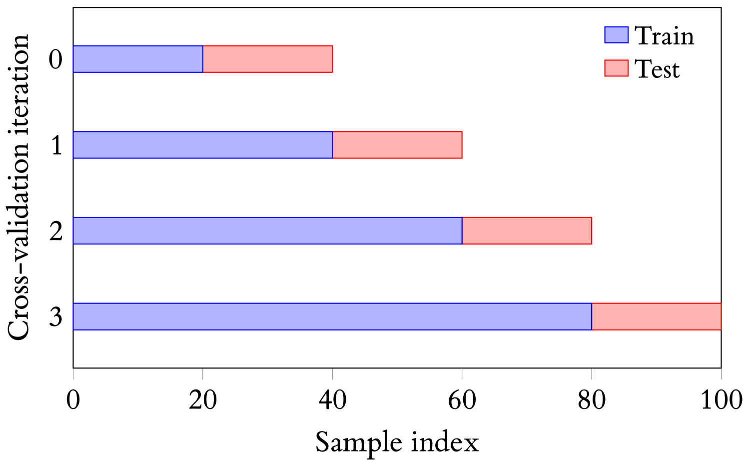

Figure 3: A visualisation of the nested time-series cross-validation behaviour that I used, after the visualisation of `sklearn.model_selection.TimeSeriesSplit`.

Model	μ	σ^2
Mean	4.43	3.82
`DecisionTreeRegressor`	3.49	2.57
`HistGradientBoostingRegressor`	3.15	2.40

(a) Separate model for each station

Model	μ	σ^2
Mean	5.45	0.06
`HistGradientBoostingRegressor`	2.60	0.08
`LGBMRegressor`	2.68	0.14
`MLPRegressor`	2.81	0.21

(b) Single model for all stations

Figure 4: The mean average error achieved by the classes of model for the cases of sub-task 1. In fig. 4a, μ is the average over stations and cross-validation folds. In fig. 4b, μ is the average over cross-validation folds only.

4 Results

4.1 Sub-task 1

References

Alin, Aylin (2010). "Multicollinearity". In: *WIREs Computational Statistics* 2.3, pp. 370–374.

Bergmeir, Christoph, Rob J. Hyndman, and Bonsoo Koo (2018). "A Note on the Validity of Cross-Validation for Evaluating Autoregressive Time Series Prediction". In: *Computational Statistics & Data Analysis* 120, pp. 70–83.

Bojer, Casper Solheim and Jens Peder Meldgaard (2021). "Kaggle Forecasting Competitions: An Overlooked Learning Opportunity". In: *International Journal of Forecasting* 37.2, pp. 587–603.

Buitinck, Lars, Gilles Louppe, and Mathieu Blondel (2013). "API Design for Machine Learning Software: Experiences from the Scikit-Learn Project". In: *ECML/PKDD 2013 Workshop: Languages for Data Mining and Machine Learning*.

Ke, Guolin et al. (2017). “LightGBM: A Highly Efficient Gradient Boosting Decision Tree”. In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc.
Pedregosa, Fabian et al. (2011). “Scikit-Learn: Machine Learning in Python”. In: *The Journal of Machine Learning Research* 12, pp. 2825–2830.