Predicting the Number of Ended Trips by Station in a Bike-Sharing System

Filipe Gonçalves
DETI
Universidade de Aveiro
Aveiro, Portugal
filipegonc@outlook.com

***Abstract*—Bike-sharing systems (BSS) are ever growing and becoming more popular by the day. Consequently, managing one is becoming more and more complex. From identifying possible problems with the bicycles, to redistribute them to their respective stations and place, from understanding the traffic flow to predict the hours of more affluence, the job of an operator of a system like this is even more difficult. The purpose of this paper is to facilitate the work of an operator, by building a predictive system based on stations and hours of a day. As such, it will be discussed different methods and compared different results, to reach a final solution.**

***Index Terms*—Bike-sharing, Bike-Sharing system, bicycle, Linear Regression, Deep Learning, feature selection, K-Fold Cross Validation, Long-Short Term Memory**

I. Introduction

BSS [1] typically operates through a network of docking stations strategically located throughout a city. These stations serve as pick-up and drop-off points for bicycles, and users can rent a bicycle from one station and return it to any other station within the system predisposed area.

The main challenges these systems have are the possible vandalism, stealing or any other malicious intent towards the bicycle; redistributing the bicycles into all stations, and knowing which stations should the bicycles be going to; and, finally, their operating system, which needs to respond in real time to any type of alerts and problems the system and any bicycle can have.

In order to try to ease the work done by any operator and the redistribution management, there is a need to build predictive systems, which will give a rough estimate of which stations should be delivered more bicycles, at each given hour.

The prediction system built and analyzed will predict the number of bicycles that will end a trip in any given hour, for any given station, and therefore any operator should be able to understand, based on these results, to which station the redistribution team should deliver all their bikes, as well as any user of this service should be able to decide to which station should it go based on the number of ended trips for each near station.

II. State of Art

There are a lot of different systems already designed and built with different frameworks, models and methods to build a predictive system.

The first work I would like to mention was published in 2016 in the 14th Annual International Conference on Mobile Systems, Applications, and Services [5], which tries to fight the redistribution problem with bicycle spatio-temporal mobility models and fine-grained traffic prediction mechanisms. It also establishes the BSS system as a dynamic network, and predicts the traffic using different features, from meteorology to time factors. They also use Random Forest Regression to combine a large enough number of decision trees built on bootstrapped samples of the dataset and take the mean of the outputs from each different decision tree to make its predictions.

With the same challenge, the work published in 2015 in the Proceedings of the 23rd SIGSPATIAL international conference on advances in geographic information systems [7], proposed a hierarchical prediction system, comprised of five components: Bipartite Station Clustering algorithm to cluster stations into groups based on geographical locations and historical transition patterns; Traffic Learning Prediction model to learn a traffic model which works as the root of the hierarchical prediction; Check-Out Proportion Learning Prediction model to allocate the traffic to each cluster, and the check-out proportion is predicted using a multi-similarity-based inference model; Inter-Cluster Transition Prediction model; Check-Out/In Inference algorithm to understand where a bike will be checked-in after being checked-out in a station at a certain time. This system tries to understand both the check-in/out of each bike as well as the duration of each trip.

Regardless of the methods used, these systems work as any other, but there are some that use a more simple and undercomplicated workaround to the problem.

The work published in 2014 on the IEEE Symposium on Computational Intelligence in Vehicles and Transportation Systems (CIVTS) [6] uses simple machine learning tools on a two year dataset to accomplish their solution. They also use very minimal features such as the week number on the calendar, and the number of bikes available in the prior hours, from one hour before until twenty four hours before. They also train and test the dataset with a variety of Regression models such as Ridge Regression [16], Adaboost Regression [17] [18], Support Vector Regression [19], Random Forest Regression [20] and Gradient Tree Boosting Regression [21].

III. Models

A. Dataset Description

The data used was created using the data from the Kaggle community free datasets, and is composed of two main projects: a general bicycle sharing system with trips made by the bicycles by hour, for 12 months [2]; and a more specific dataset with more than 700 station descriptions [3].

Each dataset contains features we used and didn't use. The main dataset with the trips by hour contains many usable features such as Hour, Temperature, Humidity, Situation of the Weather - if it's snowing, raining, sunny or cloudy -, and many more. However, the stations dataset only has the station identification and their coordinates as essential features.

We used these coordinates to map all the stations and try to create a cluster of a smaller number of stations to use in the models. We decided to group the closest stations to a certain point, which in this case is London, and gathered the resulting 40 stations with their respective identifier and saved them for our models. We can see the map with the cluster of the stations in Figure 1.

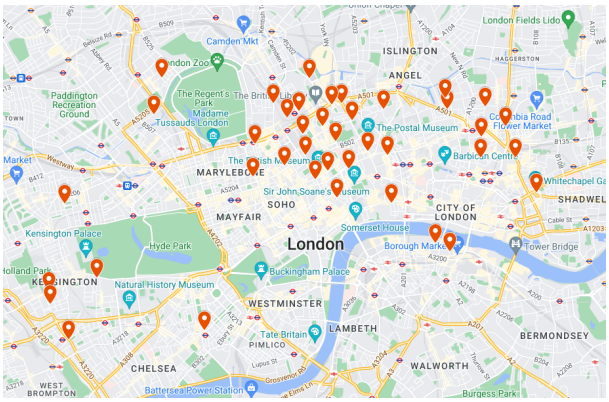

Fig. 1. Stations Cluster

Finally, we used the stations and simulated the trips into every station with a weighted probability based on closest distance to the center point of the cluster.

The final dataset contained all the features of the main dataset and the station identification, in which the user of the vehicle "ended" their trip.

B. Data Visualization

To understand which features are supposed to be baseline, and because the both base data simulated isn't collected by us and we are working with timetables, we need to evaluate the periodicity of our dataset, and understand which time span should we use for an accurate model.

As our dataset is a full year long, we decided to see if there were any periods of time, in the whole data, where we could see periodicity. As seen in Figure 2, we can't tell by only one year if there are any similarities between seasons, or time based events like christmas. The only conclusion we can accurately identify is the more we approximate ourselfs to either seasons of spring or summer, the more trips we are bound to see happening.

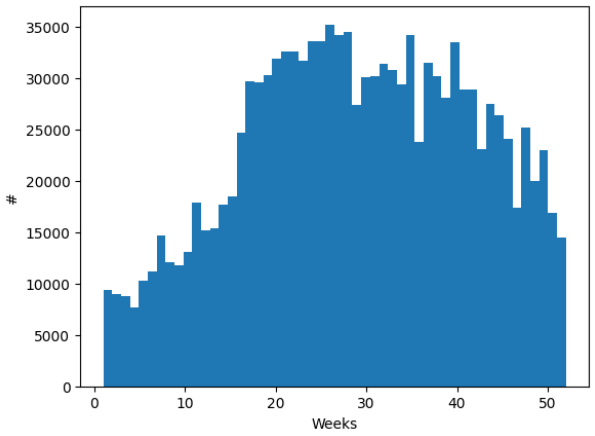

Fig. 2. Number of trips in a year by weeks

As in a year long we can't see proper periodicity, we decided to look for differences in days, by week. As a result we achieved the Figure 3. We found out that each day starts with a small decline in trips. We also understood that in normal working days, from monday to friday, we get to local maximums, one at the start of the working day and one at the end of the working day, around 7-9 hours and 17-19 hours respectively. Moreover, during the weekend we can see that the number of trips are accumulated during the full day, with a maximums near 12 and 14 hours.

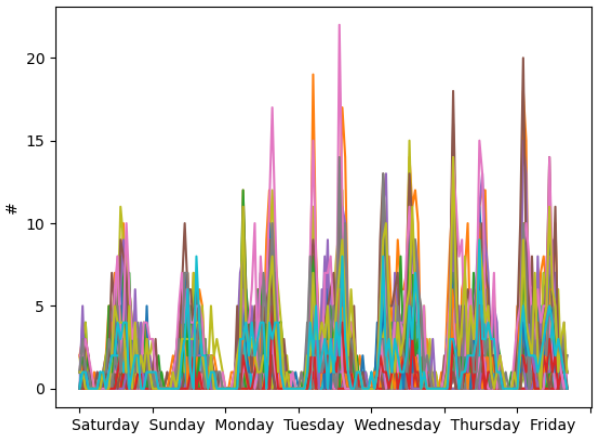

Fig. 3. Number of trips in a year by weekday

This periodicity can be explored further as seen in the Figures 4 and 5, which gives us more information about the weekdays and how the number of trips is periodic along the year.

Understanding the time periodicity in the dataset is crucial for the proposed system, but it is also vital to look into clusterization and visualize if our selected stations should work as a whole or if there is any need to create smaller clusters.

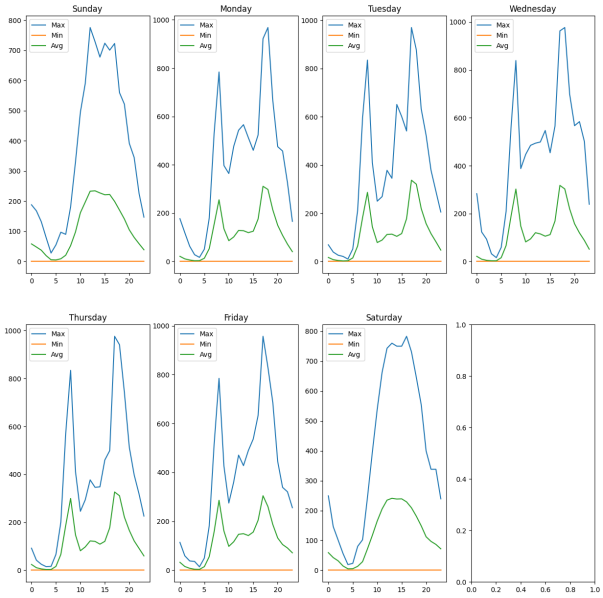

Fig. 4. Number of trips in a year by weekday by hour - separated

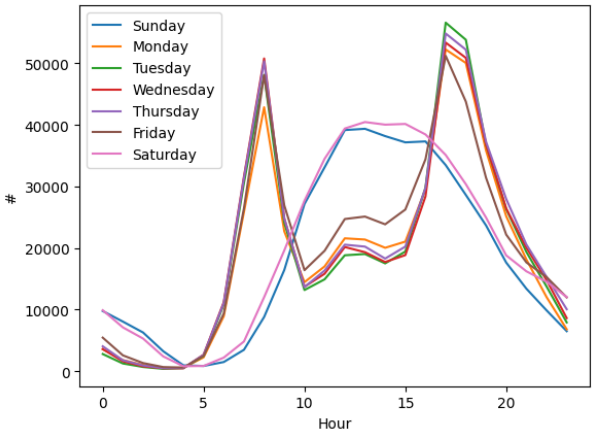

Fig. 5. Number of trips in a year by weekday by hour - together

As seen in Figure 6, there is a need to separate our stations into two clusters, which for naming purposes we shall identify them as Max Cluster and Min Cluster, for the stations with bigger number of trips and smaller number of stations, respectively.

The Max Cluster contains less than 50% of all the stations with 19 out of 40, and the Min Cluster contains the rest of them with 21 out of 40 stations.

As to which cluster should we use to train, test and validate our model, two main reasons keep us from choosing the cluster with the bigger number of trips:

- Big disparity between some stations
- Less stations

The second point is not that big of a deal, as a difference of two stations is not really relevant, however the big difference between the number of trips by two stations is greater in the

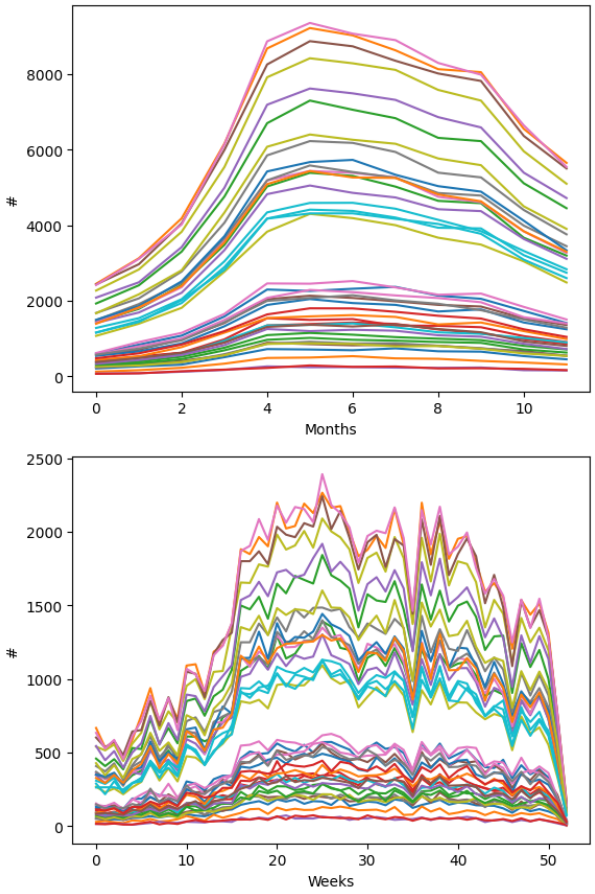

Fig. 6. Trips by each station by Months and by Weeks

Max Cluster than in the Min Cluster, which makes it less usable, and more risk imminent.

As such, we decided to use the Min Cluster for the creation of the models.

C. Regression Models

Regression [8] is a technique that should be used in any of these two theories: forecasting and prediction, whenever the application belongs to the machine learning area; in some cases, it can be used to determine relations between independent and dependent values.

Regression Models can use either theory, however Linear Regression [9] [10] is one of the most common and comprehensive machine learning algorithms, which is used to find a linear relationship between one or more predictors. Simple Linear Regression (SLR) is a model with a single independent variable [11].

As we are trying to predict only the number of trips ended in a given hour at a given station, this problem can be resolved with a SLR model.

Nevertheless, to accurately predict the result we desire, our model can't only have this little information, and to combat the problem, we decided to create different models, each with new features, as follows in Table I:

Method	Features
1	Ct(t)
2	Ct(t), Ct(t-1)
3	Ct(t), Ct(t-1), Ct(t-1week)
4	Ct(t), Ct(t-1), Ct(t-1week), Ct(t-2week)
5	Ct(t), Ct(t-1), Ct(t-1week), Ct(t-2week), Ct(t-3week)
6	Ct(t), Ct(t-1), Ct(t-1week), Ct(t-2week), Ct(t-3week), Ct(t-4week)

TABLE I
METHODS USED FOR THE REGRESSION MODELS

Given the number of trips that ended, **Ct(t)**, in a given hour, **t**, which can be the latest hour, or the latest hour the week prior, etc.

To separate the data between training, testing and validation data, we used K-Fold Cross Validation (KCV) [12], as it is a more real and validated way to split our dataset and train and test our models. KCV consists in splitting a dataset into k subsets, divided into subsets used to train, subsets used to test, and subsets used to validate the model.

However, splitting the dataset randomly is a complete mistake as we are dealing with a timetable. The solution was to create different models based on the different ways we can group our dataset. We decided to visualize the difference in these three groupings:

- Grouping based on Days
- Grouping based on Stations
- Grouping based on Trips

Grouping based on days means that we will use KFC and split our dataset based on the date of each trip, while grouping based on trips will use KFC and split our dataset based on the number of trips. While these two groupings sound the same, they have different impacts on the models created and results somewhat different.

Regardless of the way we split our data, the results for the Linear Regression models were the same, as we can see in Figure 7.

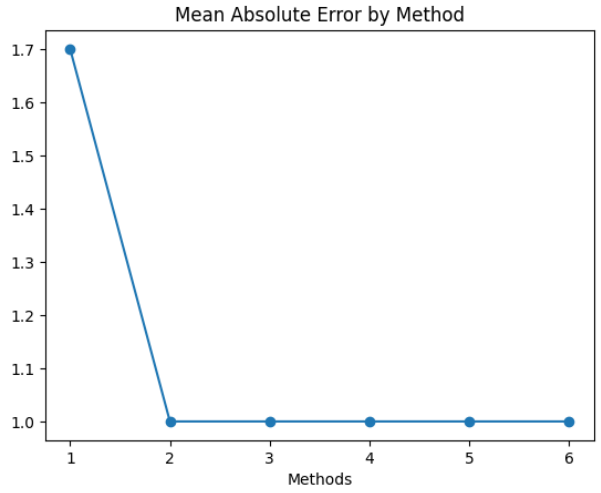

Fig. 7. Linear Regression Methods

These results were calculated while rounding the number of trips predicted, as we are working with natural numbers instead of real numbers, and calculating the score between the predicted and the true values. It was also used Mean Absolute Error (MAE) [13] as it is a much better metric in model evaluation studies, comparing to Root Mean Squared Error, or even Mean Squared Error.

Looking back at the results, we are able to understand that using only the number of trips from the hour prior, we get the worse results, averaging 1.7 trips of error, which can be approximated to almost 2 trips of error. However, looking behind any more than one week will only delay the execution of the prediction and will not give us any better results.

We can conclude that using the features mentioned for the Method 2 seems to be the most efficient and less error ocuring model.

D. Feature Extraction and Selection

Until now, there was no need to over complicate our models, which only used simple features delegated as essential. Still, to create and use the Deep Learning models, only using simple features will not get good improvements, and as such knowing which features are the best and which are the worst to implement in the models is essential.

The selected features were based on the best score each feature had with the dependent value we are trying to predict, using univariate linear regression tests returning F-statistic and p-values [14] [15]. This scoring function will rank features in the same order if all the features are positively correlated with the target.

The results were mostly the same for the different grouping styles used, with the exception of one or two features, and the mostly correlated being shown and explained in Table II.

Feature	Description	Score
Temp	Temperature (ºC)	10437
Temp Feeling	Temperature felt by humans (ºC)	10281
Humidity	Humidity percentage (%)	4554
WeatherSituation	Value to represent the seasons	1578
Windspeed	Wind speed percentage (%)	190

TABLE II
ADDITIONAL FEATURES WITH THEIR CORRELATION SCORE

There are more features we could use, however we decided to only use features with a score higher than 150, as anything below that does not have much correlation with the desired prediction.

E. Deep Learning Models

Deep learning is a technique that allows computational models to learn representations of data with multiple levels of abstraction [23]. It has not only surpassed Machine Learning models in areas like business, science and government, but it also resolved many problems derived from the artificial intelligence community for years.

With the creation of multiple processing layers, we can train our model, using this technique, to solve our problem. Moreover, as stated in the ”Supervised Sequence Labelling with Recurrent Neural Networks” [24], Long-Short Term Memory (LSTM) has proven successful at tasks requiring long

range memory, and has solved several problems that were impossible, using any other Recurrent Neural Network. With this knowledge, we decided to try and predict the number of ended trips using a LSTM layer in our model.

Similarly as before, we decided to divide our dataset, now with more features, using the same methods as before: dividing by day, stations and trips.

For each division, we created different combinations of all the usable features, which can be seen in the Tables III, IV and V.

The number of methods and the combination of features were all made based on results from previous methods, while understanding if the addition of a single feature could deteriorate the model or improve its accuracy and reducing its MAE.

Method	Features
1	Temp
2	Temp + Temp Feeling
3	Temp + Temp Feeling + Humidity
4	Temp + Temp Feeling + Humidity + WeatherSituation
5	Temp + Temp Feeling + Humidity + WeatherSituation + Windspeed
6	Temp + Temp Feeling + WeatherSituation

TABLE III
DAY DIVISION - METHODS

Method	Features
1	Temp
2	Temp + Temp Feeling
3	Temp + Temp Feeling + Humidity
4	Temp + Temp Feeling + Humidity + WeatherSituation
5	Temp + Temp Feeling + Humidity + WeatherSituation + Windspeed
6	Temp + Temp Feeling + WeatherSituation + Windspeed
7	Temp + Humidity + WeatherSituation + Windspeed
8	Temp + Humidity + Windspeed

TABLE IV
STATION DIVISION - METHODS

Method	Features
1	Temp
2	Temp + Temp Feeling
3	Temp + Temp Feeling + Humidity + WeatherSituation
4	Temp + Temp Feeling + Humidity
5	Temp + Temp Feeling + WeatherSituation

TABLE V
TRIP DIVISION - METHODS

Note that all these methods, not only contain now these features, but also the features used while evaluating Regression models, which includes features such as **Ct(t)** and **Ct(t-1)**.

The results we got from the Deep Learning methods, were not as expected, as can be seen from the Figure 8.

Using CVS with stations division is worse than using a division by trips, moreover dividing by trips is actually better than most methods from the other two separation modes.

However, as we can't say what exactly is a 1.4019 trip, as it is the MAE of one method, all these methods will be reduced as producing the same result, which is a variation of 1 trip, almost 2 trips, from the perfect result.

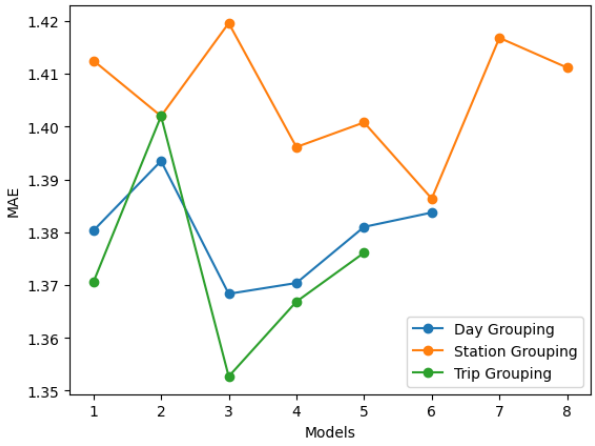

Fig. 8. Mean Absolute Error - Comparison

IV. RESULTS

Before comparing final results, we can analyze the big difference in MAE values between the best Linear Regression methods and their equivalent - using the same features - using Deep Learning techniques, as seen in Figure 9.

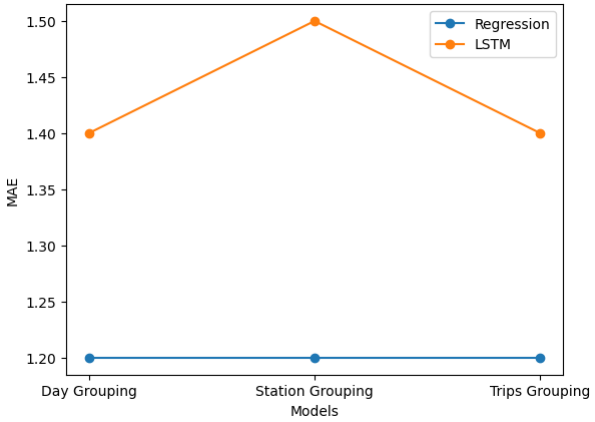

Fig. 9. Groupings - Comparison between Regression and LSTM - Regression methods

While we must understand that receiving a MAE score of 1.2 or 1.4 is essentially the same, as the difference in trips is fundamentally 1, having a low approximation to 1 is incredibly better than a high approximation to 1. Moreover, the time spent while training, testing and validating the Regression models is five times, sometimes six times, lower than the Deep Learning methods.

These two points are what fundamentally made us reach the conclusion that, for this case, with this dataset, using a Deep Learning method, or a LSTM technique is incredibly inefficient, while using the same simple features.

However, with the addition of more features and the creation of more complex methods, the later conclusions can be wrong.

Comparing the results of the best Deep Learning method, for each CVS division, and creating a Regression model with

the exact same features as the these models, we can understand which model is the best.

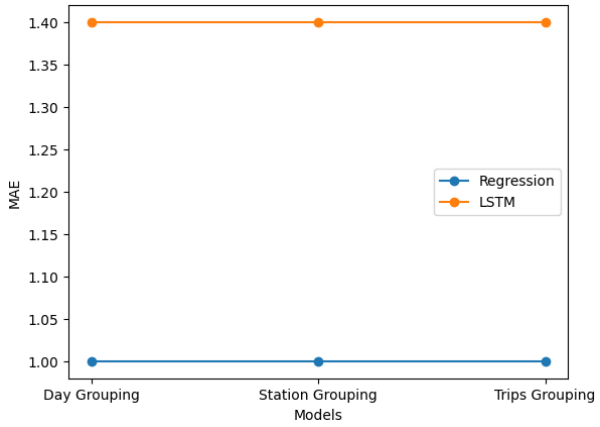

Fig. 10. Groupings - Comparison between Regression and LSTM - LSTM best methods

As we can see from Figure 10 the conclusions we took earlier are still evident. Even though, in a theoretical view these results are very different from one another - while the LSTM technique, regardless of features and CVS separation, achieved a 1.4 MAE score at best, the Regression models achieved at best a 1.0 MAE score -, in a practical manner these results mean the same - the error on number of trips each model can predict is 1.

V. Discussions

From the results we got in our experimentation and comparisons, it's obvious to tell that, in this case, Regression models are far better, not only in true results, but also in better results over less training time.

Acknowledgment

I want to thank Diogo Macedo for the inspiration, which was based mostly on his dissertation paper [4]. I would also like to thank professor Pétia Georgieva for the help and mentoring during the build and analysis of the project. Finally, I want to thank the Kaggle community for creating the base datasets for this project, as well as the authors of the library scikit-learn [14]. This study was done in the framework of the European Union-NextGenerationEU, through the National Recovery and Resilience Plan of the Republic of Bulgaria, project № BG-RRP-2.004-0005

References

[1] P. DeMaio. 2009. Bike-sharing: History, impacts, models of provision, and future. Journal of Public Transportation, vol. 12, no. 4, pp. 41–56.
[2] Akash Patel. 2021. Rental Bike Sharing Dataset. In Kaggle. https://www.kaggle.com/datasets/lakshmi25npathi/bike-sharing-dataset.
[3] Eden Au. 2018. London Bike Sharing System. In Kaggle. https://www.kaggle.com/datasets/edenau/london-bike-sharing-system-data.
[4] Diogo Macedo de Sousa. 2019. Decision Support Service for Bewegen Bike-Sharing Systems. In Universidade de Aveiro.
[5] Yang, Z., et al., 2016. Mobility modeling and prediction in bike-sharing systems. In: Proceedings of the 14th annual international conference on mobile systems, applications, and services. New York, NY: Association for Computing Machinery, 165–178.
[6] R. Giot and R. Cherrier, "Predicting bikeshare system usage up to one day ahead," 2014 IEEE Symposium on Computational Intelligence in Vehicles and Transportation Systems (CIVTS), Orlando, FL, USA, 2014, pp. 22-29, doi: 10.1109/CIVTS.2014.7009473.
[7] Li, Y., et al., 2015. Traffic prediction in a bike-sharing system. In: Proceedings of the 23rd SIGSPATIAL international conference on advances in geographic information systems. New York, NY: Association for Computing Machinery, 1–10.
[8] J. Wu, C. Liu, W. Cui, and Y. Zhang, "Personalized Collaborative Filtering Recommendation Algorithm based on Linear Regression," in 2019 IEEE International Conference on Power Data Science (ICPDS), 2019, pp. 139-142.
[9] D. Maulud and A. M. Abdulazeez, "A Review on Linear Regression Comprehensive in Machine Learning", JASTT, vol. 1, no. 4, pp. 140-147, Dec. 2020.
[10] H.-I. Lim, "A Linear Regression Approach to Modeling Software Characteristics for Classifying Similar Software," in 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC), 2019, pp. 942-943.
[11] Abdulazeez, A., Salim, B., Zeebaree, D., & Doghramachi, D. (2020). Comparison of VPN Protocols at Network Layer Focusing on Wire Guard Protocol.
[12] Anguita D, Ghelardoni L, Ghio A, Oneto L, Ridella S (2012) The 'K' in K-fold cross validation. In: Proceedings, ESANN 2012, European symposium on artificial neural networks, computational intelligence and Mmachine learning. Bruges (Belgium), 25–27 Apr 2012, i6doc.com publ.
[13] Chai, T. and Draxler, R. R.: Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature, Geosci. Model Dev., 7, 1247–1250, https://doi.org/10.5194/gmd-7-1247-2014, 2014.
[14] 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. Duch-snay, "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp.2825–2830, 2011
[15] Scikit-learn. f_regression.
[16] McDonald, G.C., 2009. Ridge regression. Wiley Interdisciplinary Reviews: Computational Statistics, 1(1), pp.93-100.
[17] 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.
[18] H. Drucker, "Improving regressors using boosting techniques," in ICML, vol. 97, 1997, pp. 107–115.
[19] 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.
[20] L. Breiman, "Random forests," Machine learning, vol. 45, no. 1, pp. 5–32, 2001.
[21] J. H. Friedman, "Stochastic gradient boosting," Computational Statistics & Data Analysis, vol. 38, no. 4, pp. 367–378, 2002.
[22] Graves, A. (2012). Long Short-Term Memory. In: Supervised Sequence Labelling with Recurrent Neural Networks. Studies in Computational Intelligence, vol 385. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-24797-2_4
[23] LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. Nature 521, 436–444 (2015). https://doi.org/10.1038/nature14539
[24] Graves, A. (2012). Long Short-Term Memory. In: Supervised Sequence Labelling with Recurrent Neural Networks. Studies in Computational Intelligence, vol 385. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-24797-2_4