2019 IEEE International Congress on Big Data (BigData Congress)

A new unsupervised predictive-model self-assessment approach that SCALEs

Francesco Ventura*, Stefano Proto*, Daniele Apiletti*, Tania Cerquitelli*, Simone Panicucci†,
Elena Baralis*, Enrico Macii‡, Alberto Macii*
* Department of Control and Computer Engineering, Politecnico di Torino, Torino, Italy
† COMAU S.p.A.
‡ Interuniversity Department of Regional and Urban Studies and Planning, Politecnico di Torino, Torino, Italy
Email: * ‡ name.surname@polito.it, † name.surname@comau.com

***Abstract*—Evaluating the degradation of predictive models over time has always been a difficult task, also considering that new unseen data might not fit the training distribution. This is a well-known problem in real-world use cases, where collecting the historical training set for all possible prediction labels may be very hard, too expensive or completely unfeasible. To solve this issue, we present a new unsupervised approach to detect and evaluate the degradation of classification and prediction models, based on a scalable variant of the Silhouette index, named Descriptor Silhouette, specifically designed to advance current Big Data state-of-the-art solutions. The newly proposed strategy has been tested and validated over both synthetic and real-world industrial use cases. To this aim, it has been included in a framework named SCALE and resulted to be efficient and more effective in assessing the degradation of prediction performance than current state-of-the-art best solutions.**

***Keywords*-Self-Assessment, Silhouette, Big Data, Industry 4.0**

I. Introduction

With the advent of the fourth industrial revolution (i.e. industry 4.0), new challenges and opportunities have been faced by Big Data and machine learning researchers. The adoption of sensors in the industry leads the companies to a completely new approach in production processes. The benefits brought by data analytics are not just relative to the production but, thanks to the huge amount of data collected by sensorized machineries, also the company decision making process is supported, bringing to more accurate and valuable results. In many real industrial contexts and applications (e.g., automotive [1]), the maintenance process of factory floors exploits complex and effective strategies (i.e. predictive maintenance) based on supervised algorithms. As widely known, supervised learning (i.e. classification) requires the historical dataset along with the labels about the events to be predicted. However, the performance of the classification task is often hard to evaluate over time: traditional supervised indexes (e.g., accuracy) require the actual class values (labels) not only for the historical data, but also for the new data, which is typically a requirement that can not be satisfied.

Prediction model performance usually degrades over time because (i) new incoming data widely differ from the data distribution on which the model was trained or (ii) because not all possible classes (labels) were effectively known at training time. To overcome this issue, new quality indexes for self-assessment are required.

This paper presents SCALE (Self-evaluating process Control AnaLytcs Engine), a scalable engine for automatic industrial equipment maintenance, where a new self-assessment strategy automatically detects when the appropriateness of a prediction model degrades too much for the data under analysis. The self-assessment strategy takes advantage of a newly proposed unsupervised and scalable index called Descriptor Silhouette (DS), able to describe the intra-class cohesion and inter-class separation, other than introducing a new degradation metric able to characterize the performances of the classification model over time when new unseen data are processed by the system.

The experimental results show that the proposed solution is able to assess the model degradation over time. Additionally, DS outperforms the current state-of-the-art strategies evaluating inter-class cohesion and inter-class separation of labeled dataset in distributed environments, thus being an innovative result in the Big Data analysis context.

The paper is organized as follows. Section II presents the current state-of-the-art for the research topic; Section III describes the methodology of the implemented engine and presents the newly proposed self-assessment strategy. Section IV shows, for a real industrial use case, the experimental results obtained by means of the presented solution and, at the end, Section V draws the conclusion of the research, focusing as well on the future direction of the work.

II. Related works

With the sensorization of factory floors, more and more machineries and production equipment are continuously collecting industrial data: as a consequence, an ever-increasing number of companies is able to collect a huge amount of information about the production processes.

The collection, the processing and the analysis of huge volumes of real-time sensor data still represents a challenge in the modern industrial context. As discussed in [2], Big Data analytics is able to face the necessity of knowledge extraction helping managers to make more-informed business decisions and to improve the production processes. Many solutions [3], [4] exploit Big Data frameworks to face the requirements

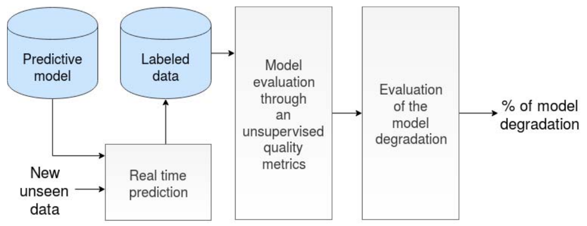

Predictive model
Labeled data
Model evaluation through an unsupervised quality metrics
Evaluation of the model degradation
% of model degradation
New unseen data
Real time prediction

Fig. 1: SCALE's methodology.

imposed by this modern productive scenario. Specifically, in [5], the authors propose a scalable architecture exploiting open source technologies (i.e. Apache Kafka and Spark) for online and offline processing along with a visualization layer. The authors in [6] present instead a Big Data analytics framework to provide a health monitoring services with application on an aerospace and aviation industrial. Moreover, in [3], an integrated Self-Tuning Engine for Predictive maintenance in Industry 4.0 is presented, taking advantage of Big Data technologies (i.e. Kafka, Spark, Cassandra) and running on top of a Docker contenerized environment. A further solution for predictive maintenance in a Big Data environment, tailored to wind turbines monitoring, has been proposed in [4] proposing a data-driven solution deployed in the cloud for predictive model generation.

The majority of the cited works does not take into account the performance degradation that the arrival of a new unknown data distributions can cause over a prediction process. To evaluate the geometrical distribution of points divided in classes (or clusters, depending on the applications) a wide number of indexes have been proposed in literature [7]. Most of them, however, are affected by a low scalability and a high computational complexity. The implementation of good and scalable evaluation metrics is still a challenge in Big Data scenarios. A scalable, approximate and aggregated Silhouette index has been recently added to the Spark MLLib library [8], named Squared Euclidean Silhouette[1] (SE-Sil). Moreover, since SE-Sil does not allow to have a score for each point in the dataset it cannot be exploited to capture model degradation.

Here we introduce a new scalable unsupervised index, named DS, to model the geometrical distribution of points in a Big Data context without requiring a-priori knowledge on data. DS has been proposed to detect when a model does not fit anymore the new input. To the best of our knowledge, no similar studies deal with an unsupervised analysis of classification model degradation. Furthermore, the Silhouette approximation provided by DS is better than the one reached by SE-Sil (see Section IV for further details).

III. Methodology

SCALE is a new general-purpose engine suited to fulfil the needs of modern companies in the context of industry 4.0. In a modern machine learning pipeline, a *Self-assessment* step is necessary to identify possible degradation of the prediction task due to changes in the production environment and decide whether it is necessary to update and retrain the predictive model with new data.

The self-assessment methodology proposed in this research, is reported in Figure 1. Given a trained *predictive model*, since its knowledge is based on the information contained in the train samples (historical sensor data with label), it is difficult to assure that its performances will remain the same over time. The direct consequence of this issue is that the *real time predictions* performed on *new unseen data* can be misleading or, worse, completely erroneous if the new data shows a previously unknown distribution. Since validation techniques require labeled data to be applied and in real use case data are usually unlabeled (the label assignment is generally a very onerous task), the common validation strategies are not applicable. To overcome this issue, the self-assessment algorithm exploits an *unsupervised quality metric* to carry out the task of *evaluating the predictive model degradation*.

The algorithm main idea is that, given a dataset of points divided in classes, the measure of the intra-class cohesion and the evaluation of the inter-class separation, before and after the prediction of unseen data, is able to detect the degradation of a predictive model. The engine can be automatically triggered when the number of incoming samples of data reach a given percentage w.r.t. the number of records in the historical dataset (e.g., 10%).

Descriptor Silhouette index. The Silhouette [9] is a well-known index with the purpose of evaluating the quality of clusters of points in terms of cohesion and separation. However, the computational cost of this index is $O(N^2)$, where N is the cardinality of the dataset (a critical dimension in a Big Data context): to calculate the Silhouette index all the pairwise distances between the points of the dataset have to be computed. Thus, the Silhouette coefficient is not suited to be applied in a Big Data application.

To solve this problem, we propose a new approach, named DS (Descriptor Silhouette), that is tunable to have a computational cost linear in the number of records and with an error with respect to the standard Silhouette score definitely acceptable.

The DS is based on the idea that the geometrical shape of a group of points can be described with a low number of descriptors, well distributed in the space. For each classification label, a certain number of descriptors can be calculated exploiting the centroids extracted through the well-known K-Means algorithm [10] (in Apache Spark MLLib [8]) to well describe its shape. The number of distances to compute can be drastically reduced, calculating just the distances between each point of the dataset and the descriptors. The computational complexity of the algorithm is now $O(N * C * D)$, where N is the cardinality of the dataset, C is the number of classes learned by the classifier, and D is the number descriptors computed for each class. The number of classes is usually very small w.r.t. the number of points and, commonly, it remains constant unless a model is retrained with new set of classes. The number of descriptors is usually several order of magnitude lower

[1]2017 - Marco Gaido (Hortonworks Inc.) - https://drive.google.com/file/d/0B0Hyo%5f%5fbG%5f3fdkNvSVNYX2E3ZU0/view

than the number of records in data collection under analysis. Indeed, K-Means and generally clustering algorithms are able to describe the dataset with a limited number of centroids, that well represent the whole geometrical points distribution. The DS approach implements the Euclidean distance and it exploits the same Silhouette definition to calculate the score for each point in the dataset.

As the Silhouette coefficient, the proposed DS index assumes values in $[-1, 1]$. The DS is computed for each record in a dataset, and it gives the information about the quality of the assignment of that record to a specific classification class. Values lower than zero represent a bad assignment of the record to a group, while values higher than zero mean a good assignment. In the specific case of model assessment, the classes of the classifier can be considered as clusters to which the new incoming unlabeled data is assigned. Thus, each classification class is described by a Silhouette curve, like the one reported in Figure 2 obtained ordering the DS values of each point in each class.

Model degradation. The *base* DS curve, obtained at the end of the model training, describes the intra-class cohesion and inter-class separation for the training dataset. Then, when new data come to the classifier it is labeled with one of the known classes. After a set of new data has been labelled, the *Self-assessment* block will recompute the DS including the new labelled points: an upwards shift of the DS curve represents an improvement in terms of intra-class cohesion and inter-class separation while, on other hand, a downwards shift denotes a degradation.

The degradation of the DS index highlights the presence of new unseen points (i.e., not present during the training of the model). This scenario can be translated into a degradation of the classification model itself: the model, indeed, is not able to recognize the new data distribution since this was not available at the time of its creation. Given a prediction model trained on a set of classes C, the degradation of a class $c \in C$ at time t is described by the following relation:

$$DEG(c,t) = \alpha * \text{MAAPE}(Sil_{t_0}, Sil_t) * \frac{N_c}{N} \tag{1}$$

$$\alpha = \begin{cases} 1 & if: \ \overline{Sil_{t_0}} \geq \overline{Sil_t} \\ -1 & if: \ \overline{Sil_{t_0}} < \overline{Sil_t} \end{cases} \tag{2}$$

The coefficient α defines if the degradation is positive (meaning a possible reduction of performances of the classification model) or negative when the new incoming data has a distribution similar to the data used to train the model increasing the cohesion of the class under analysis, and MAAPE (*Mean Arctangent Absolute Percentage Error*) [11] quantifies the curve shift. From (1), the degradation is modelled as the MAAPE error between the initial DS curve (obtained at the model creation time) and the one obtained at a certain time t, after that the model receives new data. The degradation is balanced between the classes by the ratio N_c/N, where N_c is the number of new records assigned to class c and N is the total number of new incoming data. The degradation of the whole model can be computed as the sum of the degradation for each class.

TABLE I: Washers dataset.

NumWashers	**# of cycles**	**Dataset %**
0	2,392	10.03%
1	5,367	22.52%
2	6,212	26.06%
3	8,707	36.53%
4	1,155	4.85%
Total cycles	23,833	100%

IV. Experimental results

To validate the effectiveness of the proposed methodology, all the experiments have been performed exploiting the well-known Big Data analytic framework Apache Spark [12], along with the scalable machine learning library MLLib [8]. Other useful algorithms to support the analytic process come from the *scikit-learn* machine learning library [13].

In the default configuration, the number of descriptors computed for each class and used to compute the DS is set to 4% of the cardinality of the dataset.

Experimental context. The real industrial use case on which the approach has been tested on consists in predicting the correct tensioning level of the belt installed in a robot axis based on the electricity consumed by the motor. In this use case, some washers have been used to discretize the belt tensioning levels: the higher is the number of washers, the lower the tension of the belt. The number of needed washers represents the label assigned to each cycle of measured electricity: the requirement is to predict, for each incoming electricity cycle, the right tension for the belt in terms of number of washers. Table I shows the number of cycles in the dataset under analysis, divided by class: the cycles are characterized by 5 unbalanced classes for a total of more than 20,000 samples. The best tensioning of the belt is necessary to assure the correct functioning and precision of the robot: low tension causes slippage and premature wear of the belt and the pulley, while too much tensions lead to excessive strain on belts, bearings and shafts which translates into overheating. Accordingly to domain experts, belt tensioned with 1 or 3 washers (classes 1 and 3) represent a robot functioning correctly.

Predictive model. From the experimental context for the use case under analysis, the framework presented in [3] turns out to be effective in predicting the class relative to the belt tensioning values given the electricity consumption. The framework [3] automatically selects the Random Forest, that outperforms all the other classifiers. Especially for classes 1 and 3, identified by the domain experts as the most acceptable tensioning values, the f-measure validates with a high confidence the computed models reaching an average value over a 3-fold cross-validation of over 0.9. This is the model that is used to validate the effectiveness of our approach.

Self-assessment. The real data of the industrial use case have been used to evaluate the effectiveness in measuring the degradation of the model described above.

We present two experiments based on two different models: *model_i*) a model trained on half of the class 1 and the whole class 3, and *model_ii*) a model trained on all the elements of class 1 and 3. To simulate the evaluation of the classification model over time, we used five different test sets with the following characteristics: *test_i*) a dataset composed by half of the items belonging to class 1 (not overlapped with the half used in the training of model_i and the whole class 0 with this order, *test_ii*) a set of points corresponding to the whole class 0, *test_iii*) a dataset composed by the points of class 2, *test_iv*) all the points of class 4, and *test_v*) the points belonging to classes 0, 2 and 4, ordered by class. All the test sets have been split in 10 sub-sets to simulate the evaluation at different time periods. The testing is performed with increasing sub-sets (extracted in order), containing from the 10% up to the 100% of the original cardinality, with a 10% step. From now on, the DS curve computed at the training time is indicated as t_0, while the curves computed for each split of the test sets are indicated as t_i (with i ranging from 1 to 10).

As reported in Table III, the first model, *model_i*, has been tested with the configuration *test_i*, while the model *model_ii* has been tested with the configurations *test_ii*, *test_iii*, *test_iv* and *test_v*.

The experiment ***test_i*** (Figure 3a) demonstrates that the classification of data with a known distribution (points labeled with class 1) does not affect significantly the degradation of the model, while as soon as new data with an unknown distribution comes, the degradation of the classification model starts to drastically increase. In this case, all the new incoming data (firstly belonging to class 1 and then to class 0) has been classified as class 1. Figure 3a shows this behavior over time: from time t_1 to t_5, only sensor data with real label 1 is evaluated by the trained model and the degradation remains limited between 1.87% and 9.72%. On the other hand, as soon as data from class 0 arrives, from time t_6 to t_{10}, the input is erroneously labeled by the model as belonging to class 1. Thus, the degradation score proposed in this paper is able to capture this degradation step between time t_5 and t_6, reaching 29.65% at time t_{10}.

The experiment ***test_v*** (Figures 2 and 3b) demonstrates that the proposed method is able to detect the degradation over time for multiple unseen classes. Table II reports how the new incoming data has been classified in this experiment configuration. From Figure 2 it is possible to see how the DS points out a degradation for both the classes (the new unseen data are assigned both to class 1 and to class 3). In Figure 3b it is possible to notice that from time t_1 to t_2 only data labeled as 0 comes, causing a degradation of the class 1 (degradation for class 3 is null). From time t_3 to time t_{10}, instead, as new data with real label 2 and 4 comes, this is classified as belonging to class 3, thus lowering the DS value for this class. In the same time instants, the trend of degradation for class 1 is decreasing because (1) balances the degradation w.r.t. the number of new input data accordingly to each class.

Table III shows the combinations of models and test sets along with the degradation results at time t_{10}, separately for class 1, class 3, and the total degradation. From the results, *test_ii* mostly affects class 1 while *test_iii* influences class 3. *test_iv*, on the contrary, well suits the distribution of the training set (especially for class 3), probably because it is the less represented distribution in the original dataset (only 4% w.r.t. the total number of cycles).

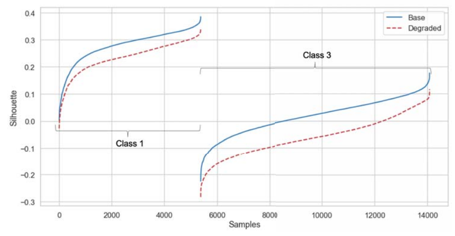
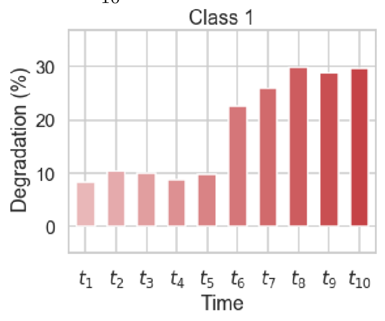
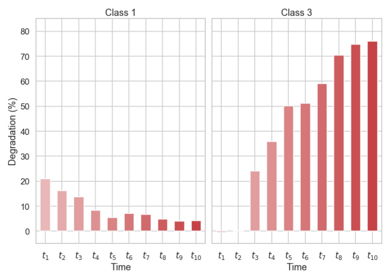


Fig. 2: DS curves for *test_v* before (Base) and after (Degraded) degradation at time t_{10}.



(a) Degradation of class 1, (unseen classes 1 and 0)



(b) Degradation of classes 1 and 3, (unseen classes 0, 2, 4)

Fig. 3: Model degradation for *test_i* (left) and *test_v* (right) at different timings.

Descriptor Silhouette performance. To prove the computation performance of the proposed approach, a synthetic dataset of 1,000,000 records with 10 features and an isotropic Gaussian distribution over 3 classes has been created exploiting the *scikit-learn* Machine learning library [13]. This dataset has been used as baseline to proof the tunable linearity of the proposed index. From this dataset, 6 sub-datasets with the same characteristics of the original one have been generated,

TABLE II: Distribution of test data (classes 0, 2 and 4) in classes 1 and 3, *test_v* configuration.

		True class		
		Class 0	**Class 2**	**Class 4**
Assigned Class	**Class 1**	2,391	4	2
	Class 3	1	6,208	1,153

TABLE III: Degradation results at time t_{10} for classes 1, 3 and total degradation.

Model	Test_set	Degr_1	Degr_3	Tot_Degr
model_i	*test_i*	29.65	0.00	29.65
model_ii	*test_ii*	12.48	0.03	12.52
	test_iii	-0.01	96.02	96.02
	test_iv	0.00	-13.56	-13.56
	test_v	4.21	75.90	80.11

TABLE IV: Descriptor Silhouette (DS) performance comparison with Standard Silhouette (SK-Sil, implemented by *scikit-learn*). DS is configured with 200 descriptors.

# records	DS_t [s]	DS_{avg}	$SK\text{-}Sil_t$ [s]	$SK\text{-}Sil_{avg}$	Err %
1,000	7.67	0.744	0.40	0.744	0.201
10,000	44.10	0.758	3.53	0.743	2.025
50,000	167	0.766	24.70	0.742	3.192
100,000	406	0.768	126	0.742	3.414
500,000	2021	0.769	2806	0.742	3.559
1,000,000	3968	0.769	11234	0.742	3.579

and for each of them we computed both the DS index and the *scikit-learn* Silhouette index. Table IV shows, for each sub-dataset, the time necessary to calculate the DS index (DS_t) and the time to obtain the standard Silhouette ($SK\text{-}Sil_t$) showing that our index is linearly proportional to the number of points in the dataset when the number of descriptors is kept constant while standard Silhouette is growing quadratically.

The average value of our DS index approximates very well the one obtained with the standard one: with 1 million points DS index is higher of just 0.027 w.r.t. the standard, meaning a good precision in evaluating the intra-class cohesion and inter-class separation. Instead, comparing the SE-Sil score (provided by the MLLib library [8]) with the average of the DS scores and the average values calculated with the standard Silhouette computed for the whole dataset we notice that the first index gives very different results w.r.t. our solution and the standard approach: even if the required time to compute the SE-Sil in the worst case is 3.15 seconds, the average score obtained with it is always higher than the standard Silhouette score of almost 0.2 and it can lead to misleading analysis.

Moreover in Table IV column *Err %* the percentage error between the silhouette curves obtained with our DS and the SK-Sil is reported, showing a very low error even between the silhouette value per document. Thus, the DS index is well suited to be used in a Big Data context and to assess the model degradation over time. Moreover, the implementation of the DS index exploits the Spark MapReduce API, ensuring the scalability in a distributed environment.

V. Conclusions and Future Works

This paper presented a new strategy to assess predictive model performances over time. It exploits an innovative and scalable index, named Descriptor Silhouette, able to identify if the new incoming data does not fit anymore the distribution on which the models were trained, so triggering a model retraining. Experiments on a real and a synthetic datasets demonstrate that DS well performs in detecting degradation of the model performances, being thus an efficient and effective approach to trigger model updates.

Future directions of this research work certainly include: (i) automatic analysis of the input data that causes a great model degradation, in order to automatically detects new labels for these data and trigger a retraining, (ii) test the newly proposed approach over other different scenarios and data collections, to furthermore prove the generality of the methodology.

Acknowledgment

The research leading to these results has received funding from the European Commission under the H2020-IND-CE-2016-17 program, FOF-09-2017, Grant agreement no. 767561 "SERENA" project, VerSatilE plug-and-play platform enabling REmote predictive mainteNAnce.

References

[1] F. Giobergia, E. Baralis, M. Camuglia, T. Cerquitelli, M. Mellia, A. Neri, D. Tricarico, and A. Tuninetti, "Mining sensor data for predictive maintenance in the automotive industry," in *5th IEEE International Conference on Data Science and Advanced Analytics, DSAA 2018, Turin, Italy, October 1-3, 2018*, 2018, pp. 351–360.

[2] S. Ren, Y. Zhang, Y. Liu, T. Sakao, D. Huisingh, and C. M. Almeida, "A comprehensive review of big data analytics throughout product lifecycle to support sustainable smart manufacturing: a framework, challenges and future research directions," *Journal of Cleaner Production*, vol. 210, pp. 1343–1365, 2019.

[3] D. Apiletti, C. Barberis, T. Cerquitelli, A. Macii, E. Macii, M. Poncino, and F. Ventura, "istep, an integrated self-tuning engine for predictive maintenance in industry 4.0," in *IEEE International Conference on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications, ISPA/IUCC/BDCloud/SocialCom/SustainCom 2018, Melbourne, Australia, December 11-13, 2018*, 2018, pp. 924–931.

[4] M. Canizo, E. Onieva, A. Conde, S. Charramendieta, and S. Trujillo, "Real-time predictive maintenance for wind turbines using big data frameworks," in *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, June 2017, pp. 70–77.

[5] G. M. D'silva, A. Khan, Gaurav, and S. Bari, "Real-time processing of iot events with historic data using apache kafka and apache spark with dashing framework," in *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT)*, May 2017, pp. 1804–1809.

[6] B. Xu and S. A. Kumar, "Big data analytics framework for system health monitoring," in *2015 IEEE International Congress on Big Data*, June 2015, pp. 401–408.

[7] Y. Liu, Z. Li, H. Xiong, X. Gao, and J. Wu, "Understanding of internal clustering validation measures," in *2010 IEEE International Conference on Data Mining*, Dec 2010, pp. 911–916.

[8] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, D. Xin, R. Xin, M. J. Franklin, R. Zadeh, M. Zaharia, and A. Talwalkar, "Mllib: Machine learning in apache spark," *J. Mach. Learn. Res.*, vol. 17, no. 1, Jan. 2016.

[9] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53 – 65, 1987.

[10] B. Bahmani, B. Moseley, A. Vattani, R. Kumar, and S. Vassilvitskii, "Scalable k-means++," *Proc. VLDB Endow.*, vol. 5, no. 7, Mar. 2012.

[11] S. Kim and H. Kim, "A new metric of absolute percentage error for intermittent demand forecasts," *International Journal of Forecasting*, vol. 32, no. 3, pp. 669–679, 2016.

[12] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, and I. Stoica, "Apache spark: A unified engine for big data processing," *Commun. ACM*, vol. 59, no. 11, Oct. 2016.

[13] 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.