International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Prediction of Web Service Performance and Successability using Comparative Analysis of Machine Learning and Deep Learning Algorithms

P. Mourougaradjane[1*], Dr. P. Dinadayalan[2]
[1]Centre for Research and Evaluation, Bharathiar University, Coimbatore – 641 046, Tamil Nadu, India.
[1]MOUROUGARADJANE@YAHOO.COM
[2]Assistant Professor, Department of Computer Science,
Kanchi Mamunivar Government Institute for Post Graduate Studies and Research,
Lawspet, Pondicherry 605008.
[2]pdinadayalan@hotmail.com

Submitted: 22/07/2022 **Accepted**: 25/09/2022

Abstract: Internet services, also known as e-services, has gained in importance as a means of providing online commercial services. Service Oriented Architecture (SOA) is built on a combination of multiple web services, each responsible for developing a specific task, in order to obtain complete professional software. Quality of Web Services (QWS) is a key characteristic for choosing a web service throughout the service configuration procedure and has a set of non-functional properties such as response time, availability, throughput, successabilty, reliability, compliance, best practices, latency and documentation is included. Now a days, Machine Learning (ML) has been used for service classification and regression problems. Though, the performance of traditional ML techniques is highly dependent on the superiority of physical feature engineering. We propose a technique to extract multiple data, procedural, and structured set metrics from a web service interface and use them as predictors to estimate QWS properties. Our proposed method applies Deep Learning (DL) methods with six dissimilar training approaches to build predictive models with successabilty rate. The outcome of the research shows that the proposed method is efficient and the investigational outcomes indicates that operational quality metrics are superior to technical and data quality metrics in terms of Mean Absolute Error (MAE), Mean Square Error (MSE), Root Mean Square Error (RMSE), Root Mean Square Logarithmic Error (RMSLE), R-Squared and Adjusted R-Squared performance metrics. The comparative study of six models concludes that Extra Trees Regressor model outperforms other five common DL training methods.

Keywords: *Deep learning, machine learning, performance metric, quality of web services, regression, successabilty.*

1. Introduction

Recently, a new technology known as web services which highlights the characteristics of web services like software, but users can use it but they don't need to own them. That is, the handler does not install the software but use them through the internet and standard rules. Communication network services and e-services have developed a significant medium on operational commercial service. Through these services an architectural illustration called Service-Oriented Architecture (SOA) is used. This method is built on combining several web services, each responsible for developing a specific task to get fully functional software [1]. The World Wide Web Consortium(W3C) has specified a common characterization for services. Services are characteristic intangible resource ability to perform responsibilities that form a context with functionality from the provider's corporate perspective requester entity. To use the service, specific service provider agent needs to be implemented. Web services that form SOA may able to perform the task within a specific time. There may be security policies that aren't available in some instances. All of these characteristics, referred to as Quality Web service (QWS), are crucial when choosing a service during the service configuration process. When it comes to implementing QWS, one of the most pressing concerns is the cost. In fact, information concerning web services is widely available. For non-functional requirements, the best service is provided to those that offer the same capabilities. Some of the non-functional properties includes response time, availability, throughput, successabilty, reliability, compliance, best practices, latency and documentation [2]. We have taken the data statistics of 2507 observations with 11 variables and considered no null values in the rows. With the statistics successabilty are predicted using the ratio between respond messages by request message. Successabilty is nothing but measuring the message without error or free from error. The user will request for certain message which means the responded message is correct or not will be determined by this successabilty ratio. This ratio will be in terms of percentage and by using the parameter of web service error free message need to be delivered. Web service detection techniques have progressed and upgraded in many ways to address today's difficulties. Numerous literature and studies have

contributed in making the detection of web services more effective and robust. This research paper proposes a regression model that predicts the success of a web service built on the parameters specified from the QWS dataset, using comparative analysis of ML and DL models. These models are assessed against several metrics and comparative investigation is performed to arrive at a robust model [3] for predicting the effectiveness of web services. The main stage of using a web service is to design a Web Service Description Language [4] (WSDL) document and then build a web service constructed on this classification. However, this prevents designers from including language-specific 2507 structures and types in their definitions. Similarly, further development of requires designers to write good advanced code for more or less simple web services. This is often tedious, time consuming, and error prone. To overcome these barriers, researchers need to develop new techniques and methods that can complicate the web service design process. The remaining papers are organized as follows: A brief description of the literature review is provided in Section II. Section III provides an overview of methodologies used, and Section IV contains detailed discussions of implementations and discussions. Section V comparative analysis and performance are conferred in this section and lastly section VI concludes the paper with future enhancements.

The key objectives of the research work are stated as follows,

- To pre-process and suppress unwanted distortions and enhancement of some important features.
- To create an information set rendered from features for the models using python libraries.
- To build models using Machine Learning and Deep Learning algorithms and train them using the rendered training information set.
- To implement various accuracy metrics and perform evaluations to achieve the best fit model for our regression problem.

2. Related Works

Several high-performance associated with QWS have been established as a result of the rapid expansion of web services. Designers are particularly troubled with QWS in internet services. Many academic researchers [5] have looked into how to predict and enhance quality of web service. Six distinct machine learning methods are used in the experimentations in this research. The web service quality was recognised and predicted from the elements of response time and throughput using Extra Trees Regressor, Random Forest Regressor, XG Boost Regressor, Hist Gradient Boosting Regressor, Bagging Regressor and Light Gradient Boosting Machine (LGBM) Regressor. It is the machine learning algorithm that rules the web services before the deep learning algorithm. We used the ML algorithm model as a evaluation to detect if the proposed model improves performance.

2.1 Extra Trees Regressor

According to the author L. Kumar et al., [6], Extra Tree Regressor (short for highly randomized T-shirt) has a different structure than traditional decision trees due to its strategy of splitting nodes. Randomly perform a divided for each maximum function, arbitrarily select a function, and choose the best split between them. If max feature is set to 1, a complete decision tree will be built every time.

2.2 Random Forest Regressor

Based on the author Kailash et al.,[7], Random Forest is a combination of several binary regression trees. The author uses an independent subset of variables to grow this large amount of binary regression trees. The dataset's bootstrap sample forms a decision tree, and Random Forest arbitrarily selects variable quantity to divide. Random forest is an example of the ensemble method used for classification purposes. To improve the performance of the excellent example, a random forest is created when using multiple decision trees containing random samples of features. For each individual tree, a new random sample is selected from the characteristics of the individual web services.

2.3 XG Boosting Regressor

In their paper Yang et al.,[8] denotes Gradient Boosting Decision Tree (GBDT) and decision tree theory are the foundations of XG Boost. It's a type of ensemble technique, namely the enhancing algorithm, that combines many Classification and Regression Trees (CART) trees to create a powerful classifier. This algorithm is used to create a tree by unceasingly adding trees and directing feature breakdown. A new function is educated each and every time a tree is added to minimise the enduring error projected by the previous model. We will have n trees once we have completed the course. In fact, we shall drop into the equivalent leaf node in each and every tree based on the properties of this sample web service. A score is assigned to each and every leaf node. Finally, all we have to do is match each tree. The total of the scores equals the predicted value of the item.

2.4 Hist Gradient Boosting Regressor

According to author Y. Yang et al., [9], Historical Gradient Boosting (HGB) regressor is a machine learning method that generates models in the form of ensembles for predictive research. GB shortens the arbitrary variances in loss functions and optimizes dissimilar layer levels. Next, a gradient expansion process was set up to enhance the cost function and repeatedly hand-picked function points. Negative direction of gradation. The HGBR method is built on an ensemble algorithm that includes a number of basic copies. Each basic model bootstraps a model from the statistics used for training and split separates the feature service into sets of regions to generate a unique tree model. Then adapt a simple model to each region.

2.5 Bagging Regressor

Xi. Li et al.,[10] determines Bagging serves as a way to improve accuracy. Therefore, use this algorithm with the WEKA tool for AWS dataset. Bagged classifiers often have much higher accuracy than the solitary classifier, which is derivative from the unique training data D. Bagging is an example of an ensemble technique. This method improves the composite classification model. This study uses the algorithm bagging with a REP tree. The REP tree is part of the decision tree derivation. DT is a tree structure like the flowchart. The bagging algorithm using the REP [11] tree is available in the WEKA tool.

2.6 Light Gradient Boosting Machine (LGBM) Regressor

W. Ke et al., [12], in their paper Light GBM aims to speed up the execution of XG Boost and decrease memory usage. Light GBM substitutes pre-sorted data structures built on the histogram process. After exploiting the histogram, there are numerous useful deceits. For instance, the bad histogram of improves the

cache hit rate (primarily since it uses leafwise). Machine learning uses sampling (by sample masses) to improve training speed for large datasets. Alternatively, you can specify sample weights all through training to associate with a particular sample type. According to author H. Wang et al. [13], Light GBM customs Gradient-based One Side Sampling (GOSS) to perform sampling algorithms. Outstanding to the histogram process, the handing out time difficulty of sparse data is not pre-sorted [14]. Also, the histogram doesn't care if the eigenvalues are 0.

3. Methodology

3.1 Dataset

The version 2.0 QWS dataset contains 2,507 web services and their QWS [15] dimensions, obtained in 2008 using the Web Services Broker (WSB) framework. Each and every tuple in this dataset characterizes the web service and its consistent 9 QWS dimensions. The first nine items are QWS metrics leisurely used by numerous web service benchmarking tools ended a 6-day period. The QWS value represents the average of the dimensions composed during this period of time. The latter 2 constraints signify the service name and a reference to the Web Service Description Language (WSDL) paper. Table 1 shows the sample AWS dataset.

The columns in the dataset are:

• **Response_ Time:** Period of time for sending a request and receiving a response (milliseconds)

• **Availability:** Total amount of successful views divided by Total number of views (%)

• **Throughout:** Total number of calls in a specific time period (calls / second)

• **Success:** Number of responses / numbers of request messages (%)

• **Reliability:** Ratio of the number of error messages to the total number of messages (%)

• **Compliance:** The percentage of the WSDL document that observes with the WSDL specification.

Table 1. AWS DATASET With nine metrics

First rows

	Response_Time	Availability	Throughput	Successability	Reliability	Compliance	Best_Practices	Latency	Documentation	Service_Name
0	63.25	98	25.6	100	67	78	66	10.25	30	SIAP
1	140.50	97	18.0	99	73	78	84	50.00	40	PipelineManagerALE
2	167.50	86	16.1	86	73	78	84	2.00	91	CalcService
3	287.22	86	5.5	86	53	89	66	77.57	7	AnalysisWSAppLabImplServ
4	244.00	86	26.7	95	83	100	91	2.00	9	HelloOpenSOAP
5	1138.33	88	4.7	88	73	100	80	402.33	11	FindBlankWS
6	154.00	98	14.0	99	73	100	80	3.00	4	BaseformService
7	1226.00	91	6.3	97	73	100	84	3.00	12	Personalizer
8	55.50	81	19.1	82	73	100	80	4.75	3	AmazonHistoricalPricing
9	287.22	85	3.5	86	53	89	66	101.05	5	AnalysisWSAppLabImplServ

• **Best_ Practices:** The range (%) that the web service trails the WSI undeveloped profile.

• **Latency:** The time it takes the server to develop a particular demand (in milliseconds)

• **Document:** Degree of document in WSDL (ie, descriptive tag) (%)

• **Service_ Name:** The title of the web service

• **WSDL_ Address:** The position of the WSDL (Web Services Definition Language) folder on the web.

3.2 Removing Nullity by column

The dataset contains 2507 rows and determine there is no null values in the rows. It is must that all the rows are non-null values. We are going to predict successabilty. Successabilty is the ratio between number of respond message to the number of requested messages. Successabilty is the metric of measuring the message without error or free from error [15]. Here, user request certain message means the responded message is correct or not will be predicted and determined by this ratio. This ratio will be in terms of percentage. Through the given web service parameter, we will predict the error prone message.

4. Implementation and Results

This section comprises of the research results and discussions. The proposed research involves six top models such as Extra Tree Regressor, RF, XGB, Hist Gradient Boosting Regressor, Bagging Regressor and LGBM Regressor.

4.1 System Architecture

Fig 1 demonstrates the system architecture of the proposed method. Total of 1643 web services available from open resource used for the experiment. Top 6 regression models [16] are selected to find the consistency of the responded message. We take sample as already existing web service data with all the needed reading and check for the consistency prediction of error prone messages.

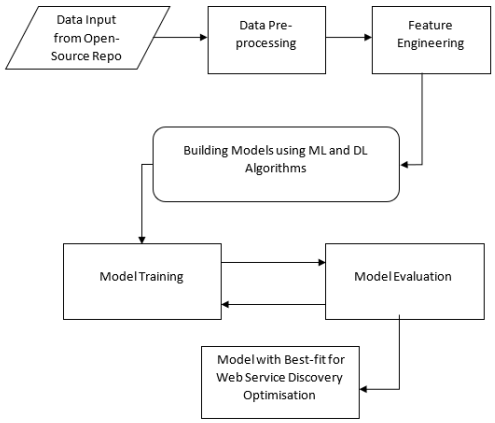

Fig 1. Proposed System Architecture

4.1.1 Data Pre-processing

Web services relies heavily on data pre-processing. It is an extremely complicated operation that accounts for 80% [17] of the whole cleaning process. Because log files include noisy, irrelevant, and unclear data, pre-processing is required. The purpose of data preparation is to improve data quality and accuracy during the prediction. Data cleansing, field extraction, user identification, and session identification are all aspects of pre-processing. The major responsibilities in this research are data cleaning and extracting individual user behaviour. As a result, we can describe the preceding task in two steps: (1) Clean up the weblog and eliminate redundant data and (2) Saves individual user behaviour. The next step data WSDL columns are

removed. Since, the data are hyperlinked with services which we don't consider in our model.

4.1.2 Label Encoder

In Python label encoding replaces the category value with a number obtained by subtracting 1 from the number of classes from 0. Use (0, 1, 2, 3, 4 and 5) if the categorical variable value contains six different classes. In this work, service name is in string which is non-numerical. Label encoding is used for converting string value into numerical.

4.1.3 Model Testing and Training

After label encoding, the default services are split into X and Y which are taken from target to predict the successabilty. We split the feature of 25% and target to 75% train. Once train is splitted, the data are standardized into percentage. Since some of the scaling will be in milliseconds but maximum scaling will be in percentage. In order to avoid scaling issue, all the data are standardized as percentage [18].

4.1.4 Lazy Predictors

Lazy predict is very laidback and spontaneous for everyone familiar with scikit-learn. First generate an instance of the estimator (Lazy Classifier in this case) and then use the Fit method to fit the data. Get predictions from all models for each individual observation by specifying predictions = True when creating the Lazy Classifier instance. Normally, we set library and load the model and will train them. Instead by doing them manually, lazy predictors will load the model, train and test the data and will sent the data to all the models and iterate to form an output data frame model will be shared. Table 2 shows the lazy predicts of top six models with the metrics R squared, Adjusted R squared, Root Mean Squared Error (RMSE) [19] and time taken.

Table 2. Lazy Predictors for Top 6 models

Model	Adjusted R-Squared	R-Squared	RMSE	Time Taken
ExtraTreesRegressor	1.00	1.00	1.07	0.20
RandomForestRegressor	1.00	1.00	1.33	0.40
XGBRegressor	1.00	1.00	1.33	0.10
HistGradientBoostingRegressor	1.00	1.00	1.36	0.56
BaggingRegressor	1.00	1.00	1.38	0.04
LGBMRegressor	1.00	1.00	1.42	0.08

4.1.5 Machine Learning Algorithms

The top six models such as Extra Tree Regressor, RF, XGB, Hist Gradient Boosting, Bagging and LGBM are compared normally then each and every top six models default parameters are processed manually. To validate this model, six different error metrics are used and examined. Extra Tree Regressor is the best fit algorithm as RMSE is less when compared with all other models. Similar to Random Forest, the ET algorithm arbitrarily service feature at each division point of the decision tree. Contrasting Random Forest, which uses a greedy system to select the best division point, the ET algorithm randomly selects the division point. The ET algorithm uses traditional top-down methods to build an ensemble of unpruned DT or regression trees. The two key variances from other tree-based ensemble techniques are that the nodes are split by randomly selecting intersections, and the entire training service is used to grow the tree. Therefore, there are three major hyperparameters that need to be tuned by the algorithm. These are the amount of DT in the ensemble, the quantity of input services arbitrarily selected and considered for each break fact, and the lowest quantity of web service samples required by the node to generate a new break point. It has two parameters. X, which is the number of attributes randomly selected for each node, and Y, which is the lowest model size for splitting the node. The total quantity of trees is represented by M. Random selection of dividing points rises the variance of the algorithm, but the decision trees are less correlated within the ensemble. This rise in variance can be addressed by cumulating the number of trees used in the ensemble. Parameters X, Y, and M have diverse effects. X regulates the strong point of the quality web service, Y determines the strength of the average output noise, and M determines the strength of the variance decrease of the ensemble model accumulation.

5. Performance Metrics

The experimental results are correlated and compared with the six different error metrics which are determined by the best fit model Extra Trees regressor model.

5.1 Correlations

The correlations of different parameters of the 2507 rows of web services are compiled with the parameter's response time, availability, throughput, successabilty, reliability, compliance, best practices, latency and documentation which are shown in Fig 2. X and Y axis denotes parameters and the calculation are based on matrix. The string data is converted into numerical values.

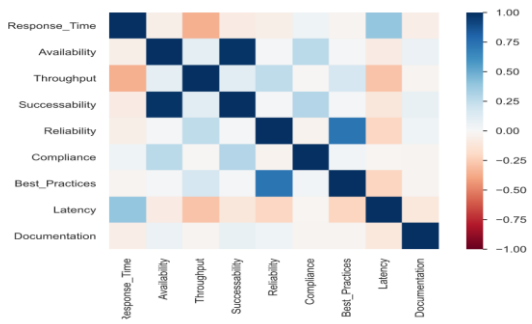

Fig 2. Correlations of Wb services parameters

5.2 Missing Values

There are no missing values and the data does not contain null values. All the rows are non-null values demonstrated in Fig 3.

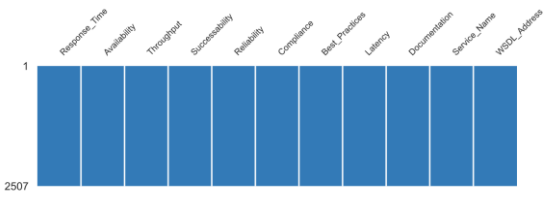

Fig 3. Missing Values and row without null values

5.3 Successabilty

Successability is the ratio between number of respond message with the number of requested messages. Measuring the message without error or free from error is the need of successabilty. The success rate is calculated with the response time taken are shown

in Fig 4. Successability measures in terms of percentage (%) and response time is measured in terms of seconds(s)

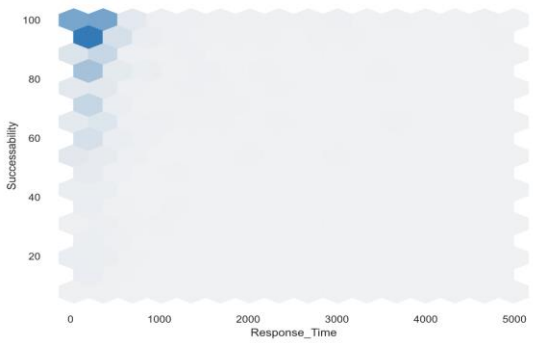

Fig 4. Successabilty Parameters

5.4 Lazy predict of Six Models

Lazy predict of six top models are determined and evaluated using the metrics R Squared, Adjusted R squared, RMSE and time taken are shown in Fig 5. R squared and adjusted R-Squared are measured in terms of second (s) and RMSE and time taken denoted by second(s)

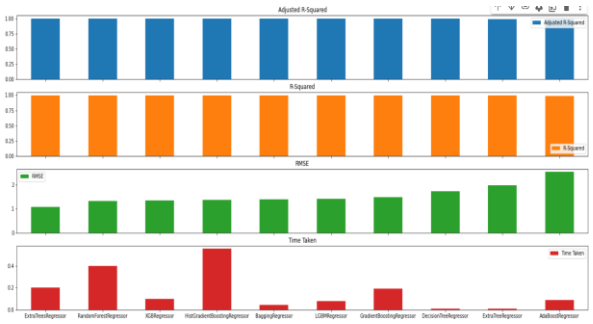

Fig 5. Lazy Predict of six Models

5.5 Error Metrics

Table 3 depicts the top 6 models' comparative analysis and error metric

Table 3. Top six Model Comparative Analysis with Error metrics

MODEL	MAE	MSE	RMSE	RMSLE	R SQUARED	ADJUSTED R SQUARED
Extra Tree Regressor	0.563955342902711 3	1.184213716108453	1.088215840772616 6	0.084539511821499 85	0.997299990691598	0.997260606439125 4
Random Forest Regressor	0.642743221690590 3	1.718452631578947 2	1.310897643440916	0.270712126554983 67	0.996081925045963 5	0.996024773223295 2
XGB Regressor	0.841500121060360 8	1.779299409166777 8	1.333903823057261 5	0.288109848235618 47	0.995943194288467	0.995884018840486 8
HistGradient Boosting Regressor	0.775038300333956 5	1.854970625328870 2	1.361973063363908 9	0.308934430265895 4	0.995770663785537 9	0.995708971685164 9
Bagging Classifier	0.668899521531100 4	1.919904306220095 6	1.385606115106344 1	0.326137672143621 8	0.995622614881484 3	0.995558763234698 8
LGBM Regressor	0.814820927783834 6	2.011611325723251 7	1.418312844799500 4	0.349468027753125 8	0.995413522719371 9	0.995346621105878 1

Mean Absolute Error (MAE) is a measure of the error between pairs of observations that represent the same spectacle shown in Fig 6. MAE is represented in terms of second(s) Examples of Y versus X include predicted and observed times, subsequent and initial time comparisons, and measurement and alternative measurement methods. MAE is calculated as shown in eqn (1)

$$MAE = \sum \frac{yi-xi}{n} \quad (1)$$

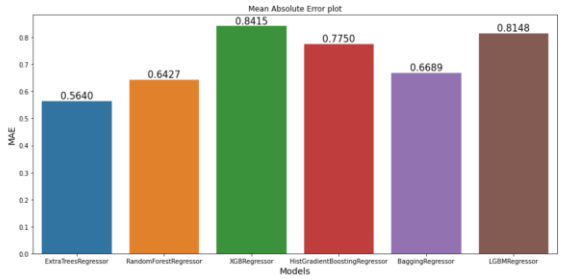

Fig 6. Mean Absolute Error

The MSE is a metric that assesses the accuracy of an estimate of the web services. This will give a positive value that drops as the error lines zero as it is consequent from the square of Euclidean distance. MSE plot are denoted by the term seconds (s) which are shown in Fig 7.

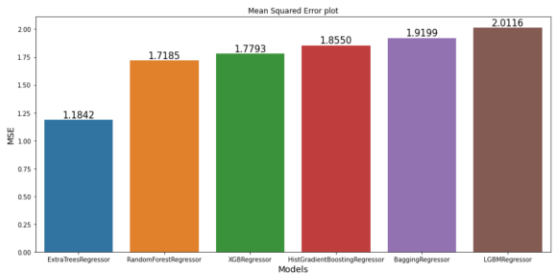

Fig 7. Mean Squared Error

The square root of MSE is Root Mean Squared Error is denoted by the term seconds (s) are shown in Fig 8. The logarithmic value of RMSE is RMSLE is denoted by the term seconds (s) are shown in Fig 9.

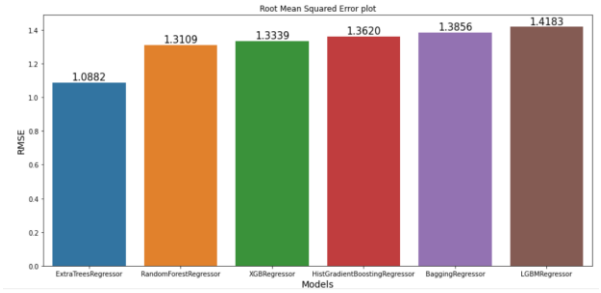

Fig 8. Root Mean Squared Error

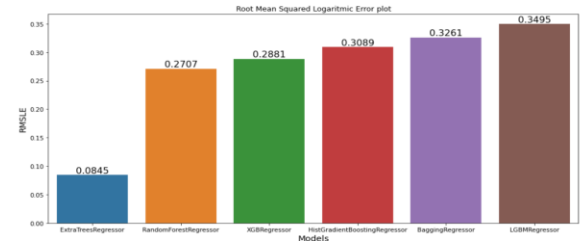

Fig 9. Root Mean Squared Logarithmic Error

R SQUARED is the square of R is denoted by the term seconds (s) are shown in Fig 10. Where R is the correlation coefficient calculated.

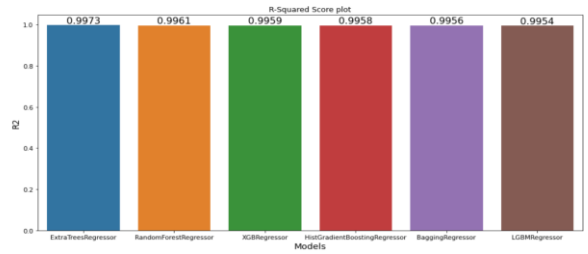

Fig 10. R Squared Score

ADJUSTED R SQUARE is the term taken by adjusting the R correlation value is denoted by the term seconds (s) are shown in Fig 11.

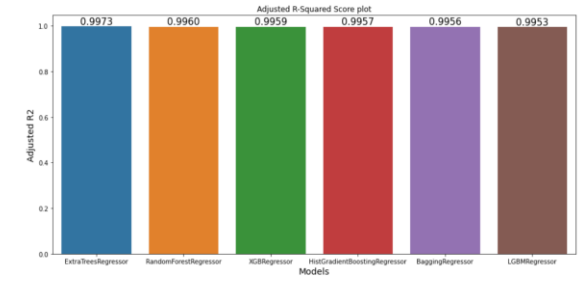

Fig 11. Adjusted R Squared Score

Attributes used are the R squared value set. Regression output model is formed and predicted using R squared. Here, R squared and Adjusted R squared are equivalent. Through the table 3 if R squared decreases automatically RMSE score will get increases. If RMSE get increases MAE, MSE increases. Thus, Extra Tree Regressor is the model which are best fit for successability through the default parameters and lazy predict techniques.

6. Conclusion and Future Scope

A robust model assessed for a variety of precision metrics that use ML and DL algorithms to forecast the productivity of a particular web service beneath numerous parameters and predict the initial failure of the web service. Web services permit different requests to interconnect with each other and exchange data and services with each other. Numerous works and investigate efforts contribute to making the discovery of Web services more efficient and robust. This research paper proposes a regression model that predicts the success of a web service based on specified parameters using comparative analysis of ML and DL models. The model is evaluated against numerous metrics and comparative analysis is performed to arrive at a robust model for predicting the efficiency of web services. Regression output models are built and predicted using the coefficient of determination. Here, R-Squared and Adjusted R-Squared are equivalent. Table 3 shows that the RMSE score automatically increases as the coefficient of determination decreases. As RMSE increases MAE, MSE increases. Therefore, Extra Tree Regressor is the best model for success with default parameters and delay prediction techniques. The limitations are only 1600 web services are used. For the future scope many models with new techniques can be incorporated for the better successabilty.

References

[1] Y. Yang, W. Ke, W. Wang, and Y. Zhao, “Deep Learning for Web Services Classification,” *2019 IEEE International Conference on Web Services (ICWS)*, pp. 440-442, 2019. doi: 10.1109/ICWS.2019.00079.

[2] Ling Guo, Ping Wan, Rui Li, Gang Liu, and Pan He, “Runtime Quality Prediction for Web Services via Multivariate Long Short-Term Memor,” Article in Mathematical Problems in Engineering, August 2019. https://doi.org/10.1155/2019/2153027.

[3] D. Chen, M. Gao, A. Liu, M. Chen, Z. Zhang, and Y. Feng, “A Recurrent Neural Network Based Approach for Web Service QoS Prediction,” 2019 2nd International Conference on Artificial Intelligence and Big Data (ICAIBD), pp. 350-357, 2019. doi: 10.1109/ICAIBD.2019.8837006.

[4] Garg, D. K. . (2022). Understanding the Purpose of Object Detection, Models to Detect Objects, Application Use and Benefits. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 8(2), 01–04. https://doi.org/10.17762/ijfrcsce.v8i2.2066

[5] M. Chippa, A. Priyadarshini, and R. Mohanty, “Application of Machine Learning Techniques to Classify Web Services,” 2019 IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS), pp. 1-7, 2019. doi: 10.1109/INCOS45849.2019.8951339.

[6] Y. Feng, M. Gao, and Z. Zhang, “Web Service QoS Classification Based on Optimized Convolutional Neural Network,” 2019 IEEE 14th International Conference on Intelligent Systems and Knowledge Engineering (ISKE), pp. 584-590, 2019. doi: 10.1109/ISKE47853.2019.9170368.

[7] Jat, N. C., and C. . Kumar. “Design Assessment and Simulation of PCA Based Image Difference Detection and Segmentation for Satellite Images Using Machine Learning”. International Journal on Recent and Innovation Trends in Computing and Communication, vol. 10, no. 3, Apr. 2022, pp. 01-11, doi:10.17762/ijritcc.v10i3.5520.

[8] L. Kumar, and A. Sureka, “Neural network with multiple training methods for web service quality of service parameter prediction,” 2017 Tenth International Conference on Contemporary Computing (IC3), pp. 1-7, 2017. doi: 10.1109/IC3.2017.8284307.

[9] Kailash Chander Bhardwaj, and R. K. Sharma, “Machine learning in efficient and effective web service discovery,” Journal of Web Engineering, vol.14, pp. 196–214, July 2018.

[10] Yang Song, "Webservice reliability prediction based on machine learning," Journal on Computer Standards and Interfaces, vol.73, pp. 103466, 2021. ISSN 0920-5489. https://doi.org/10.1016/j.csi.2020.103466.

[11] Paithane, P. M., & Kakarwal, D. (2022). Automatic Pancreas Segmentation using A Novel Modified Semantic Deep Learning Bottom-Up Approach. International Journal of Intelligent Systems and Applications in Engineering, 10(1), 98–104. https://doi.org/10.18201/ijisae.2022.272

[12] Y. Yang, N. Qamar, P. Liu, K. Grolinger, W. Wang, Z. Li, and Z. Liao, "Serve Net: A Deep Neural Network for Web Services Classification. IEEE International Conference on Web Services (ICWS), pp. 168-175, 2020. doi: 10.1109/ICWS49710.2020.00029.

[13] Y. Yang, X. Li, W. Ke, and Z. Liu, "Automated prototype generation from formal requirements model," IEEE Transactions on Reliability, vol.69, no. 2, pp. 632–656, 2020.

[14] Y. Yang, X. Li, Z. Liu, and W. Ke, "RM2PT: A tool for automated prototype generation from requirements model," in Proceedings of the 41th International Conference on Software Engineering: Companion Proceedings (ICSE'19), pp. 59–62, May 2019.

[15] Y. Yang, W. Ke, and X. Li, "RM2PT: Requirements validation through automatic prototyping," in Proceedings of 27th International Requirements Engineering Conference (RE'19), pp. 484–485, Sep. 2019.

[16] H. Wang, L. Wang, Q. Yu, Z. Zheng, and Z. Yang, "A proactive approach based on online reliability prediction for adaptation of service-oriented systems," Journal of Parallel and Distributed Computing, vol.114, pp. 70–84, 2018.

[17] H. Wang, Z. Yang, Q. Yu, T. Hong, and X. Lin, "Online reliability time series prediction via convolutional neural network and long short-term memory for service-oriented systems," Knowledge-Based Systems, 159, pp. 132–147, 2018.

[18] X. Sun, S. Wang, Y. Xia, and W. Zheng, "Predictive-trend aware composition of web services with time-varying quality-of-service," IEEE Access, vol.8, pp. 1910–1921, 2020.

[19] X. Xu, S. Fu, and L. Qi, "An IoT-Oriented data placement method with privacy preservation in cloud environment," Journal of Network and Computer Applications, vol.124, pp. 148–157, 2018.

[20] André Sanches Fonseca Sobrinho. (2020). An Embedded Systems Remote Course. Journal of Online Engineering Education, 11(2), 01–07. Retrieved from http://onlineengineeringeducation.com/index.php/joee/article/view/39

[21] W. Chen, B. Liu, H. Huang, S. Guo, and Z. Zheng, "When UAV swarm meets edge-cloud computing: the QoS perspective," IEEE Network, vol.33, no. 2, pp. 36–43, 2019.

[22] L. Qi, X. Zhang, S. Li, S. Wan, Y. Wen, and W. Gong, "Spatial temporal data-driven service recommendation with privacy preservation," Information Sciences, vol.515, pp. 91–102, 2020.

[23] M. Hasnain, M. F. Pasha, I. Ghani, B. Mehboob, M. Imran, and A. Ali, "Benchmark Dataset Selection of Web Services Technologies: A Factor Analysis," in IEEE Access, vol.8, pp. 53649-53665, 2020. doi: 10.1109/ACCESS.2020.297925