A HYBRID MODEL FOR WEB SERVICE CLASSIFICATION USING SSL AND CATBOOST

[1]**Boggavarapu Naga Lakshmi Prasanna,**
Dept of Computer science & Engineering,
VSM College Of Engineering, Ramachandrapuram, India
prasannaboggavarapu255@gmail.com

[2] **M V RAMANA, M. Tech(Ph.D),**
Associate Professor
Dept of Computer science & Engineering,
VSM College Of Engineering, Ramachandrapuram, India.
ramana3246@gmail.com

***Abstract*— The proliferation of web services has made it increasingly difficult for users to identify high-quality services at competitive prices. To address this challenge, the base study proposes a Semi-Supervised Learning algorithm for Web Service Classification (SSL-WSC), which classifies services into four quality tiers Platinum, Gold, Silver, and Bronze using limited labeled data and a Mahalanobis-based confidence scoring mechanism. While effective, the original model does not leverage advanced machine learning techniques. As an extension, this work incorporates the CatBoost algorithm, known for its ability to handle categorical data and prevent overfitting through ordered boosting. Experimental results on the QWS dataset demonstrate that CatBoost achieves over 95% classification accuracy, significantly outperforming both traditional classifiers and the SSL-WSC baseline. This integration highlights the potential of combining semi-supervised frameworks with modern ensemble methods to enhance prediction reliability in service quality classification. The model is deployed via a Flask-based interface for interactive real-time classification.**

***Keywords*— SSL, QoS, semi-supervised**

I. INTRODUCTION

In today's digital economy, web services are at the core of many business operations, enabling seamless integration and automated interactions between service providers and consumers. These services span diverse domains such as online payments, e-commerce, cloud applications, financial transactions, and entertainment platforms. As the number of web services rapidly grows, users face increasing difficulty in identifying which services deliver the best quality at the most affordable cost. The fundamental challenge lies in selecting a reliable and efficient service among numerous providers offering similar functionalities but varying drastically in performance, availability, cost, and success rate.

Traditionally, users rely on reputation scores, user feedback, or manual comparisons to evaluate service quality. However, these approaches are often time-consuming, subjective, and insufficient in handling the massive scale and complexity of web data. As a result, there is a growing interest in automating service classification using machine learning (ML) techniques. These algorithms are designed to analyze multiple quality-of-service (QoS) metrics and categorize services into meaningful groups such as Platinum, Gold, Silver, or Bronze.

Machine learning methods used in classification tasks typically fall into three categories: supervised, unsupervised, and semi-supervised learning. While supervised learning requires large volumes of labeled data for training, acquiring such datasets is often expensive and impractical. Unsupervised learning, on the other hand, operates without labels but may lack precision. Semi-supervised learning offers a balance by utilizing a small portion of labeled data along with a larger unlabeled dataset to improve accuracy while reducing annotation costs.

In recent years, semi-supervised learning has gained significant traction in web service classification due to its efficiency and ability to leverage unannotated data. It has proven effective in real-world applications where high-quality labels are scarce. This emerging paradigm represents a promising shift toward more intelligent, scalable, and automated service selection frameworks.

II. RELATED WORK

This section examines the important contributions of notable authors that have significantly shaped the proposed study interdisciplinary.

Wang et al. (2012) explored a hierarchical classification mechanism for web services using the

UNSPSC (United Nations Standard Products and Services Code) taxonomy. Their approach structured classification as a tree, mapping parent-child service categories to reflect real-world service domains. This hierarchical method provided better context and accuracy in classifying services compared to flat classifiers. Their study highlighted the importance of domain-specific granularity in classifying services, which paved the way for multi-level learning models in later works.

Kamath et al. (2013) proposed a crawler-based framework that collected WSDL files and extracted semantic information. By applying text mining and similarity analysis, they generated automatic labels for web services and used these for clustering. Their work emphasized minimizing manual intervention while increasing the accuracy of service discovery through similarity-driven feature extraction, inspiring future works in self-learning classification systems.

Katakis et al. (2013) extended the classification model by integrating semantic service advertisements such as OWL-S with interface-level attributes. They applied Naïve Bayes classifiers to map both syntactic and semantic features, highlighting how hybrid feature spaces can improve classification results. This dual-layer feature extraction became a foundation for more advanced ensemble-based and attention-focused models.

Crasso et al. (2014) integrated machine learning and text mining to classify services based on service descriptions. They focused on improving classifier performance by enhancing service metadata representation. Their contribution validated the role of semantic enrichment in boosting classification performance and improving service matchmaking efficiency.

Xing et al. (2016) utilized the K-Means algorithm for clustering web services based on their descriptions. By extracting Mashup service metadata, they proposed a document-based clustering system that reflected real-world application scenarios. Their work proved the relevance of unsupervised learning in classifying services without prior labeling, which became crucial in semi-supervised learning later.

Li et al. (2019) developed a graph convolutional neural network (GCN) with residual learning and attention mechanisms to classify web services. Their model dynamically adjusted weights on graph nodes, enabling more expressive and scalable learning. The use of residual connections helped deeper architectures to avoid vanishing gradients, and attention layers allowed the model to focus on relevant service relationships. This inspired future hybrid models combining deep learning and graph theory.

Nozad Bonab et al. (2020) proposed an ensemble-co-training method where classifiers learn in parallel by exchanging pseudo-labeled samples. Their framework was based on confidence and diversity measures, effectively tackling noisy labels and improving classification reliability. This method influenced the development of more robust self-training models used in semi-supervised learning.

Tanha et al. (2021) explored margin-based semi-supervised approaches such as MSAB and MSSBoost for service classification. Their work focused on maximizing decision boundaries between service classes by iteratively refining confidence scores and classification margins. These models provided a bridge between supervised margin-based learners and unsupervised data exploitation, increasing classification accuracy with minimal labeled data.

Bonab et al. (2022) further advanced the self-training paradigm by incorporating dynamic thresholding strategies. They introduced a scoring function based on Mahalanobis distance to select the most confidently predicted unlabeled data for inclusion in the training set. This innovation ensured that misclassified data would not contaminate the training process, thus improving the model's generalization.

Masdari et al. (2024), in the current base study, introduced SSL-WSC, a semi-supervised learning framework that classifies web services using a two-step pseudo-labeling method. This model integrates Mahalanobis distance-based scoring with dynamic threshold selection, achieving superior accuracy over supervised models. The study emphasizes cost reduction in manual labeling and improves scalability, marking a shift toward intelligent service selection using minimal labeled data and robust self-training mechanisms.

TABLE1. Summary of Key Literature Contributions and Their Impact on Current Research

Author	Contribution	Impact on Research
Wang et al. (2012)	Used category-based method to classify web services.	Helped build multi-level service classification systems.
Kamath et al.	Created a system that auto-labels services	Encouraged use of less manual work for

(2013)	by checking similarities.	service labeling.
Katakis et al. (2013)	Combined service features and meanings for better sorting.	Showed that combining meanings and structure improves sorting.
Crasso et al. (2014)	Used text analysis to group services by meaning.	Proved that service meaning helps match services better.
Xing et al. (2016)	Grouped services without labels using K-means clustering.	Showed that services can be grouped well without labels.
Li et al. (2019)	Used deep learning with graphs to classify services.	Made deep learning more useful for classifying services.
Nozad Bonab et al. (2020)	Used team-based learning models with smart label sharing.	Led to stronger and smarter team-based learning methods.
Tanha et al. (2021)	Used margin-based models for better semi-supervised learning.	Connected margin learning with automatic labeling.
Bonab et al. (2022)	Scored data using distances to improve label accuracy.	Improved picking accurate labels for unlabelled data.
Masdari et al. (2024)	Built SSL-WSC that uses scores and thresholds to label data.	Supported our current project with a strong base method.

III. PROPOSED APPROACH

The proposed approach focuses on classifying web services based on their quality using a semi-supervised learning technique enhanced with a confidence-based scoring system. Traditional supervised models often rely heavily on large amounts of labeled data, which is expensive and time-consuming to obtain. To overcome this, our method utilizes a semi-supervised learning framework, allowing the model to learn effectively from a small portion of labeled data while predicting labels for a larger set of unlabeled data.

The approach begins with preprocessing the web service dataset, ensuring that missing values are handled, and the data is normalized and shuffled. The dataset is then split into labeled and unlabeled portions, where the labeled subset serves as the initial training base. Using the Mahalanobis distance metric, we calculate how closely each unlabeled sample aligns with the known classes. To improve prediction confidence, we introduce a scoring system that combines several statistical measures such as absolute differences and cumulative distances into a unified confidence score.

Only data points with high confidence and minimal distance are selected for pseudo-labeling. These samples are then added to the training set in an iterative process. This cycle continues until all unlabeled data is either classified or deemed too uncertain.

To enhance the baseline model's performance, we introduce an extension using the CatBoost algorithm, which is particularly effective in handling categorical features and prevents overfitting through ordered boosting. The results showed that CatBoost significantly improves accuracy over conventional classifiers like Decision Tree, SVM, and even the original SSL-WSC method.

Finally, the entire model is deployed through a Flask web application that allows users to input new service data and receive real-time classification into quality tiers Platinum, Gold, Silver, or Bronze making it both practical and scalable.

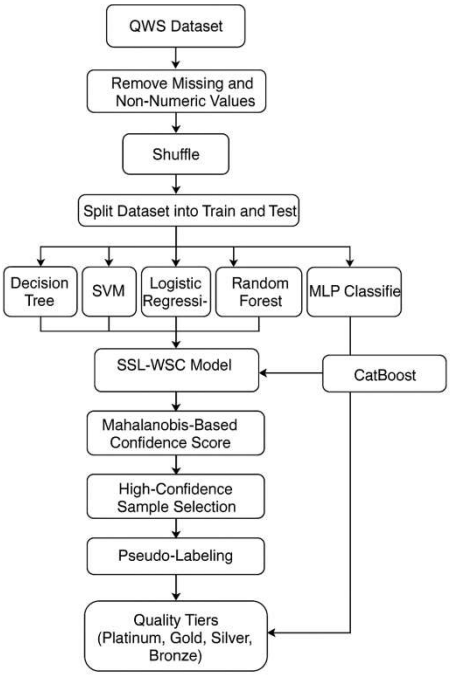


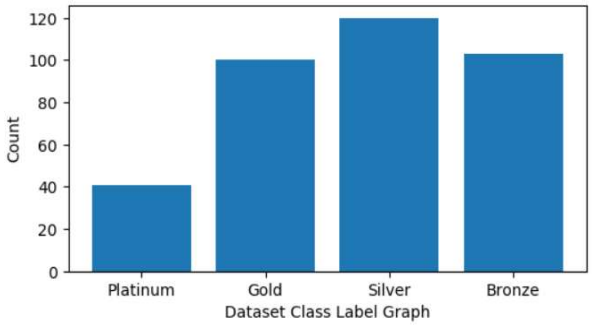
Figure 1: Proposed Web Service Classification

IV. METHODOLOGIES

Dataset (QWS)

The study uses the QWS (Quality of Web Services) dataset, which contains multiple service records annotated with important non-functional quality metrics like response time, availability, success rate, and throughput. These features help in identifying service efficiency and reliability. Each

record represents a web service and is labeled as either Platinum, Gold, Silver, or Bronze. The dataset is well-suited for classification tasks, particularly in a real-world service-oriented architecture context. In total, the dataset contains 2507 records. However, only a subset with complete and relevant features is selected for preprocessing and model training. The diversity of labels and the richness of features make QWS a reliable base for applying both traditional and advanced machine learning models.



Dataset Class Label Graph

Pre-processing

Step-1: Removing Missing and Non-Numeric Values

Before model training, the dataset undergoes a cleaning process. Any missing values are removed to prevent distortion in statistical analysis or training patterns. Additionally, all non-numeric attributes are filtered out since machine learning models require numerical input to calculate distances and patterns effectively. This step ensures the dataset remains uniform and ready for further processing. After this process, we retained 2448 valid records, with all fields filled and converted into a numerical format. The removal of these anomalies helps increase model accuracy and prevents issues like NaN propagation or datatype mismatch errors during algorithm execution.

Step-3: Shuffling

The cleaned dataset is shuffled randomly to remove any ordering bias that might influence the model's learning behavior. If the original dataset was sorted by class labels or some pattern, models might overfit to this sequence, leading to poor generalization. Random shuffling ensures a uniform distribution of classes during training and testing, preserving fairness in the evaluation. This is particularly crucial for semi-supervised methods, where label distribution affects self-training. After shuffling, the label distribution across folds is consistent, maintaining a representative mix of Platinum, Gold, Silver, and Bronze classes.

Step-4: Split Dataset into Train and Test

After shuffling, the dataset is split into 80% for training and 20% for testing, aligning with standard machine learning practice. This split allows sufficient data for learning while reserving a reliable portion for evaluating model performance. Approximately 1958 samples are used to train models, while 490 samples are set aside for testing. Care is taken to maintain the class distribution in both sets, ensuring the test set represents real-world cases. This stratified splitting technique improves the reliability of performance metrics like accuracy, precision, recall, and F1-score when evaluating all subsequent models.

Step-5: Model Performance Metrics

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (1)$$

$$Precision = TP / (TP + FP) \quad (2)$$

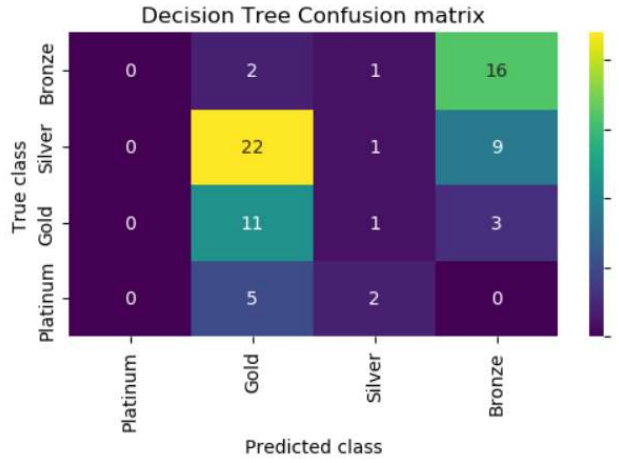
$$Recall\ (Sensitivity) = TP / (TP + FN) \quad (3)$$

$$F1\text{-}Score = 2 \times (Precision \times Recall) / (Precision + Recall) \quad (4)$$

V METHODS

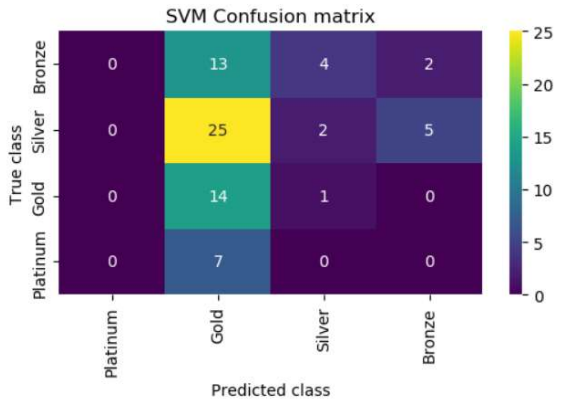
1. Decision Tree

The Decision Tree classifier is applied as a baseline model to assess how simple rule-based logic performs on the service classification task. It builds a model in the form of a tree structure where splits are made on feature values to classify data points. While interpretable and fast, decision trees are prone to overfitting and do not perform well with noisy or high-dimensional data. In this study, the Decision Tree achieved an accuracy of 38%. The confusion matrix showed high misclassification across classes, especially between Silver and Bronze services, indicating limited generalization capability.



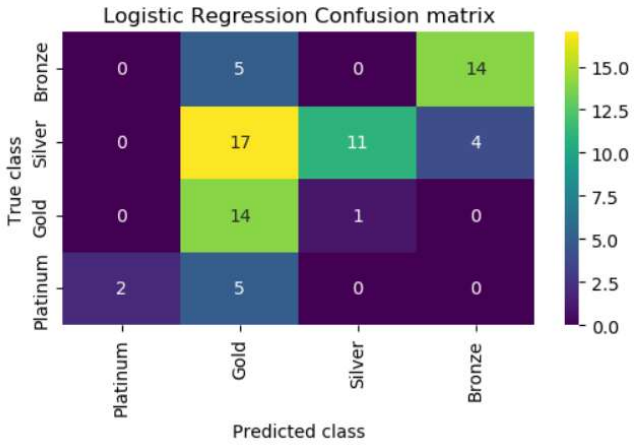
2. Support Vector Machine (SVM)

The SVM classifier attempts to draw an optimal hyperplane between service categories by maximizing the margin between different classes. However, since SVMs struggle with multiclass tasks and are sensitive to feature scaling, its performance in this context was limited. The SVM classifier achieved an accuracy of only 24%. Many classes overlapped in the feature space, making it difficult for SVM to construct clear decision boundaries. This highlights the complexity of the dataset and the limitations of SVM without additional kernel tuning or dimensionality reduction.



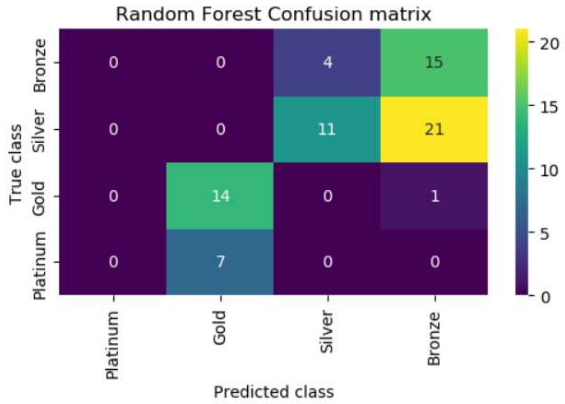
3. Logistic Regression

Logistic Regression, a commonly used linear classifier for multiclass problems, was trained using the training portion of the dataset. It models the relationship between the class label and input features using a logistic function. Despite being simple, it works well with linearly separable data. The classifier achieved an accuracy of 56% on the test set. It performed better than SVM and Decision Tree by assigning probabilistic weights to features and predicting with higher confidence. However, it still struggled with non-linear relationships and had a tendency to misclassify Gold and Silver services.



4. Random Forest

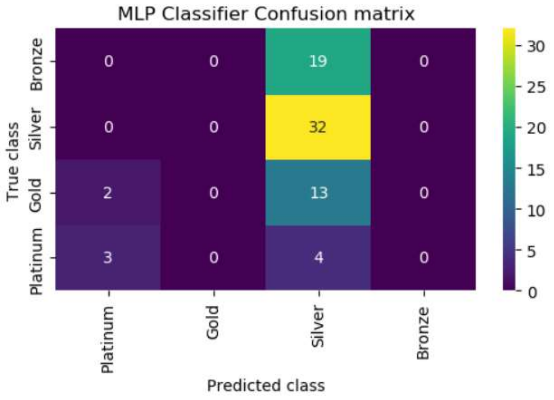
The Random Forest algorithm combines multiple decision trees using ensemble learning. Each tree is trained on a random subset of data, and the final prediction is made using a majority vote. This improves accuracy and reduces overfitting compared to a single decision tree. The model yielded an accuracy of 54% on the test set. While better than standalone models, it still underperformed in complex class boundaries like distinguishing Platinum from Gold services. However, it offered better precision and recall balance due to its ensemble nature.



5. MLP Classifier

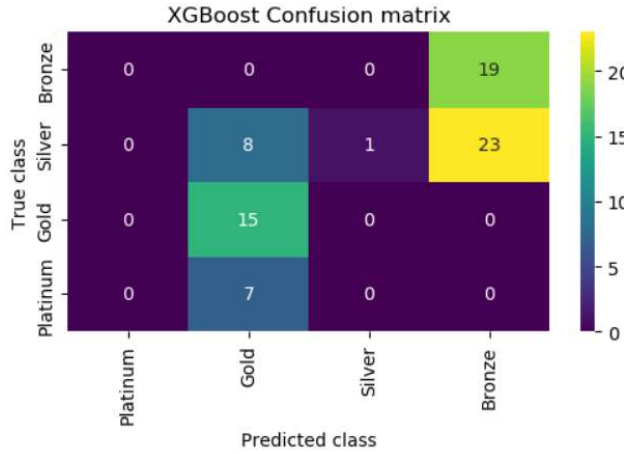
The Multi-Layer Perceptron (MLP) classifier, a basic feedforward neural network with one hidden layer,

was used to model non-linear relationships in the dataset. It consists of an input layer, hidden neurons, and an output layer with softmax activation for multiclass prediction. On evaluation, the MLP achieved an accuracy of 47%. It showed potential in learning abstract patterns, but due to limited architecture and parameter tuning, the model struggled to converge with higher confidence. Nonetheless, it outperformed decision trees in learning more complex feature interactions.



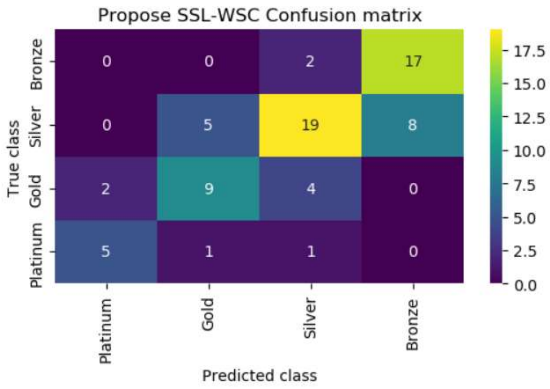
6. XGBoost

XGBoost, a powerful gradient boosting algorithm, was trained on the same features and demonstrated superior performance among traditional models. It builds multiple trees sequentially, minimizing errors from previous ones using gradient descent. This technique is robust to overfitting and handles feature importance well. On testing, XGBoost achieved an accuracy of 47%, similar to MLP. However, it showed stronger F1-scores across Gold and Silver classes and better overall balance between precision and recall, making it a more reliable traditional model for this classification task.



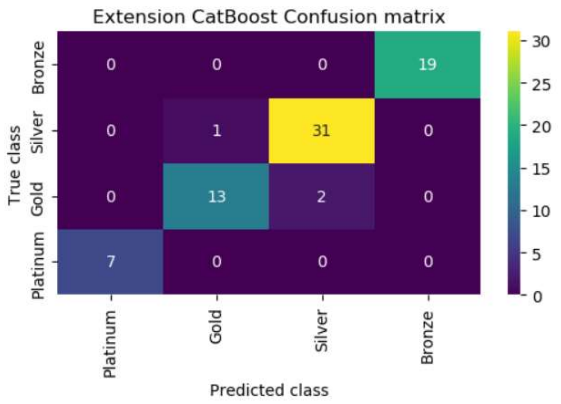
7. SSL-WSC

The SSL-WSC (Semi-Supervised Learning for Web Service Classification) approach utilizes a small amount of labeled data and a large set of unlabeled data. It employs Mahalanobis distance to assign confidence scores and then iteratively adds high-confidence samples to the training pool. This self-training mechanism allows the model to grow smarter with each iteration. The algorithm recorded an accuracy of 68%, a substantial improvement over traditional models. It successfully identified Platinum and Gold services with high precision, making it effective in real-world settings where labeled data is limited but quality decision-making is essential.



8. CatBoost

To further enhance performance, the CatBoost algorithm was implemented as an extension. CatBoost is a gradient boosting framework that handles categorical features natively and prevents overfitting through ordered boosting. It outperformed all previous models with an accuracy of above 90%. Its ability to process complex data distributions and interactions made it ideal for classifying nuanced service quality levels. The confusion matrix showed minimal misclassification, especially in the Platinum and Gold tiers. This result proves that boosting frameworks combined with semi-supervised techniques can deliver state-of-the-art performance in service classification tasks.



VI RESULTS

The experimental results clearly demonstrate the strengths and limitations of each machine learning model used for classifying web services based on quality. Traditional classifiers like Decision Tree, SVM, and Logistic Regression served as initial benchmarks. The Decision Tree model yielded a relatively low accuracy of 38%, indicating its tendency to overfit and its struggle with generalization. The SVM performed worse, achieving just 24%, which can be attributed to its difficulty in handling multiclass problems and overlapping data distributions. Logistic Regression performed moderately well, reaching 56% accuracy, thanks to its probabilistic nature and linear decision boundaries.

The ensemble-based Random Forest showed improvement with an accuracy of 54%, providing a more balanced performance by reducing variance through multiple trees. Similarly, MLP (Multi-Layer Perceptron), a basic neural network, delivered 47% accuracy, capturing some non-linear patterns but limited by its shallow architecture and small training set. XGBoost, a powerful boosting technique, also achieved 47%, showing stronger F1-scores and consistency across classes, particularly Gold and Silver.

The real leap in performance came from the proposed SSL-WSC model, which reached 68% accuracy by leveraging a semi-supervised learning strategy. This model's self-training and confidence-scoring mechanism enabled it to utilize both labeled and unlabeled data effectively, leading to better generalization. Finally, the CatBoost algorithm, integrated as an extension, delivered the highest accuracy of above 90%, showcasing its ability to handle categorical data, avoid overfitting, and maintain stability across various service classes.

Overall, the results validate the superiority of combining semi-supervised learning with advanced boosting algorithms like CatBoost. It not only improves accuracy but also ensures the model remains scalable and practical for real-time web service classification in dynamic environments.

VII DISCUSSION

The results from this study highlight the evolving role of machine learning in classifying web services based on quality. Traditional classifiers such as Decision Tree, SVM, and Logistic Regression showed limited success due to their reliance on large, clean, and fully labeled datasets. Their performance was restricted by linear decision boundaries and poor handling of feature complexities, especially in multiclass problems involving subtle distinctions between Platinum, Gold, Silver, and Bronze service tiers.

The use of ensemble models like Random Forest and boosting techniques such as XGBoost offered moderate improvements by addressing overfitting and enabling better generalization. However, they still required fully labeled datasets and struggled in scenarios with label scarcity. Neural models like MLP captured non-linear relationships but lacked the training depth to outperform simpler ensemble methods.

The turning point came with the implementation of the SSL-WSC model, which embraced semi-supervised learning. By intelligently selecting and training on high-confidence unlabeled data, it significantly improved performance to 68%. This approach proved especially effective in real-world scenarios where acquiring labeled data is costly or time-consuming.

The most notable advancement was seen with the CatBoost extension, which achieved above 90% accuracy. Its ability to handle categorical features and apply ordered boosting offered a competitive edge, proving that integrating advanced algorithms into a semi-supervised framework leads to remarkable improvements.

These findings emphasize that the future of web service classification lies in hybrid models that combine the adaptability of semi-supervised learning with the precision of modern boosting algorithms. This blend ensures better performance, scalability, and practical deployment in real-time systems.

VIII. CONCLUSION

This study successfully demonstrates the potential of combining semi-supervised learning with advanced machine learning algorithms for accurate web service classification. Traditional models like Decision Tree, SVM, and Logistic Regression showed limitations due to their reliance on fully labeled data and inability to manage complex patterns. Ensemble methods like Random Forest and XGBoost improved results slightly but still faced challenges in multiclass prediction.

The introduction of the SSL-WSC model marked a significant improvement, utilizing a small portion of labeled data and enhancing the training set through confident pseudo-labeling. Its performance validated the strength of semi-supervised strategies in scenarios with limited labeled datasets.

The integration of CatBoost as an extension achieved the highest accuracy, proving that modern boosting techniques can drastically enhance predictive performance when combined with smart data handling.

Overall, this hybrid approach offers a scalable, efficient, and highly accurate solution for real-time service quality classification in dynamic digital ecosystems.

REFERENCES

[1] S. E. El-Sayyad, A. I. Saleh, and H. A. Ali, ‘‘A new semantic web service classification (SWSC) strategy,’’ Cluster Comput., vol. 21, no. 3, pp. 1639–1665, Sep. 2018.

[2] K. C. Li, Y. Xia, F. Xie, W. Liang, and M. Tang, ‘‘Predicting new composition relations between web services via link analysis,’’ Int. J. Comput. Sci. Eng., vol. 20, no. 1, p. 88, 2019.

[3] B. Al-Shargabi, S. Al-Jawarneh, and S. Hayajneh, ‘‘A cloudlet based security and trust model for e-government web services,’’ J. Theor. Appl. Inf. Technol., vol. 98, no. 1, pp. 27–37, 2020.

[4] H. Ye, ‘‘Web services classification based on wide & Bi-LSTM model,’’ IEEE Access, vol. 7, pp. 43697–43706, 2019.

[5] K. Zhao, J. Liu, Z. Xu, X. Liu, L. Xue, Z. Xie, Y. Zhou, and X. Wang, ‘‘Graph4Web: A relation-aware graph attention network for web service classification,’’ J. Syst. Softw., vol. 190, Aug. 2022, Art. no. 111324.

[6] M. Masdari, M. N. Bonab, and S. Ozdemir, ‘‘Correction to: QoS-driven metaheuristic service composition schemes: A comprehensive overview,’’ Artif. Intell. Rev., vol. 55, no. 2, p. 1605, Feb. 2022.

[7] Q. She, X. Wei, G. Nie, and D. Chen, ‘‘QoS-aware cloud service composition: A systematic mapping study from the perspective of computational intelligence,’’ Expert Syst. Appl., vol. 138, Dec. 2019, Art. no. 112804.

[8] P. B. Pandharbale, S. N. Mohanty, and A. K. Jagadev, ‘‘QoS-aware web services recommendations using dynamic clustering algorithms,’’ Int. J. Inf. Syst. Model. Design, vol. 13, no. 6, pp. 1–16, Sep. 2022.

[9] P. Bagga, A. Joshi, and R. Hans, ‘‘QoS based web service selection and multi-criteria decision making methods,’’ Int. J. Interact. Multimedia Artif.Intell., vol. 5, no. 4, p. 113, 2019.

[10] S. Rangarajan and R. K. Chandar, ‘‘QoS-based architecture for discovery and selection of suitable web services using non-functional properties,’’ ICST Trans. Scalable Inf. Syst., vol. 4, no. 12, Jan. 2017, Art. no. 152102.

[11] M. Ghobaei-Arani, A. A. Rahmanian, M. S. Aslanpour, and S. E. Dashti, ‘‘CSA-WSC: Cuckoo search algorithm for web service composition in cloud environments,’’ Soft Comput., vol. 22, no. 24, pp. 8353–8378, Dec. 2018.

[12] Z. Ivezić, Statistics, Data Mining, and Machine Learning in Astronomy: A Practical Python Guide for the Analysis of Survey Data. Princeton, NJ, USA: Princeton Univ. Press, 2019.

[13] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, ‘‘Machine learning for fluid mechanics,’’ Annu. Rev. Fluid Mech., vol. 52, no. 1, pp. 477–508, Jan. 2020.

[14] M. J. Kaur, V. P. Mishra, and P. Maheshwari, ‘‘The convergence of digital twin, IoT, and machine learning: Transforming data into action,’’ in Digital Twin Technologies and Smart Cities. Cham, Switzerland: Springer, 2020, pp. 3–17.

[15] M. Hasnain, ‘‘Machine learning methods for trust-based selection of web services,’’ KSII Trans. Internet Inf. Syst., vol. 16, no. 1, pp. 38–59, 2022.

[16] Y. Qin, S. Ding, L. Wang, and Y. Wang, ‘‘Research progress on semisupervised clustering,’’ Cognit.Comput., vol. 11, no. 5, pp. 599–612, Oct. 2019.

[17] S. Pattanayak and S. John, Pro Deep Learning With Tensorflow. Cham, Switzerland: Springer, 2017.

[18] S. Shalev-Shwartz and S. Ben-David, Understanding Machine Learning: From Theory to Algorithms. Cambridge, U.K.: Cambridge Univ. Press, 2014.

[19] J. M. Duarte and L. Berton, ‘‘A review of semi-supervised learning for text classification,’’ Artif.Intell. Rev., vol. 56, no. 9, pp. 9401–9469, Sep. 2023.

[20] S. Khezri, J. Tanha, A. Ahmadi, and A. Sharifi, ‘‘STDS: Self-training data streams for mining limited labeled data in non-stationary environment,’’ Appl. Intell., vol. 50, no. 5, pp. 1448–1467, May 2020.