International Journal on Science and Technology (IJSAT)
E-ISSN: 2229-7677 ● Website: www.ijsat.org ● Email: editor@ijsat.org

Browser-Based Customer Churn Prediction using WebAssembly (Pyodide)

Zulkifl-Uddin Khairoowala[1], Raman Ravindra Bole[2], Gaddam Madhuprasad[3], Abhinash Naik[4], M. Somanath[5]

[1]Assistant Professor, Parul University, Vadodara, India
Email: zulkifl.khairoowala40439@paruluniversity.ac.in
[2,3,4,5]CSE, Parul University, Vadodara, India
[1]2203031250116@paruluniversity.ac.in,
[2]2203031250027@paruluniversity.ac.in, [3]2203031250109@paruluniversity.ac.in,
[4]2203031250060@paruluniversity.ac.in

Abstract

Customer churn is a major challenge for businesses, as retaining customers is generally far more cost-effective than acquiring new ones. Traditional churn prediction systems depend on server-side machine learning, creating challenges such as data privacy risks, network latency, and infrastructure complexity. This study presents the first systematic evaluation of WebAssembly (Pyodide) for fully in-browser churn prediction, eliminating the need for server-side processing. Five algorithms were evaluated—Logistic Regression, Random Forest, Support Vector Machine, Neural Network, and XGBoost—across three benchmark datasets from the telecommunications domain. Results show that browser-based models retain 85–92% of server-side accuracy while ensuring complete data privacy and reducing infrastructure overhead by 40–60%. Logistic Regression and Random Forest achieved the best trade-off between accuracy and efficiency, with cross-browser testing confirming feasibility even on mobile devices. This work demonstrates that in-browser churn prediction is a viable, cost-effective alternative to traditional deployments, with strong implications for privacy-preserving analytics in SMEs and regulated industries.

Keywords: Customer Churn Prediction, Machine Learning, WebAssembly, Pyodide, Edge Computing, Privacy-Preserving Analytics

1 Introduction

Customer retention represents a critical strategic imperative, particularly in competitive markets where acquisition costs typically exceed retention investments by five- to twentyfive-fold. Churn prediction is therefore essential for organizational profitability and longterm sustainability. The telecommunications industry exemplifies this challenge, with annual churn rates exceeding 20–30%, resulting in billions in lost revenue. Similar patterns are observed in subscription services, e-commerce platforms, and financial services, where customer lifetime value is closely tied to retention effectiveness. Predictive analytics leveraging machine learning enables organizations to identify at-risk customers and implement proactive retention interventions, such as targeted marketing, personalized offers, and improved customer service.

1.1 Problem Statement

Current churn prediction systems rely heavily on server-side machine learning, introducing several limitations:

- **Data Privacy and Compliance:** Server-side processing requires transmitting customer data, raising privacy concerns and regulatory challenges under GDPR, CCPA, HIPAA, and similar frameworks.
- **Network Latency:** Real-time predictions require immediate responses to user interactions. Server-side inference incurs network delays, affecting timely interventions.
- **Deployment Complexity and Cost:** Server-side ML demands infrastructure, load balancing, model versioning, and monitoring, increasing operational overhead.
- **Scalability Bottlenecks:** Peak loads or rapid growth can overwhelm server resources, necessitating costly auto-scaling mechanisms.

1.2 Research Gap and Opportunity

WebAssembly (WASM) and Pyodide enable near-native performance for in-browser machine learning. Pyodide allows execution of Python-based ML libraries (NumPy, Pandas, Scikitlearn) directly in the browser, without server infrastructure.
Browser-based processing addresses these server-side limitations through several key mechanisms: :

- Keeping customer data on client devices, mitigating privacy risks.
- Eliminating network delays for immediate predictions.
- Simplifying deployment via standard web infrastructure.
- Reducing operational costs by removing server computation.

However, challenges remain, including cold start latency, memory limits, browser compatibility, and reduced computational power.

1.3 Research Questions

This study investigates four key questions:

1. **Accuracy Parity:** How does in-browser predictive accuracy compare to server-side models?
2. **Performance Characteristics:** What are the computational metrics (loading time, inference speed, memory use)?
3. **Practical Limitations and Advantages:** What deployment benefits and constraints exist for in-browser churn prediction?
4. **Algorithm Feasibility:** How do different ML algorithms perform under browser constraints?

1.4 Research Contributions

This work makes the following contributions:

- **Comprehensive Evaluation:** First systematic assessment of WebAssembly (Pyodide) for browser-based churn prediction.
- **Multi-Algorithm Benchmarking:** Comparison of five ML algorithms on three telecom datasets.
- **Server vs. Browser Framework:** Comparative analysis of accuracy, latency, memory, and compatibility.
- **Privacy and Scalability Analysis:** Trade-off evaluation between privacy, performance, and scalability.
- **Deployment Guidelines:** Recommendations for algorithm selection and performance thresholds.

2 Related Work

2.1 Customer Churn Prediction Using Machine Learning

Recent advances in churn prediction highlight the effectiveness of ensemble methods and deep learning across industries. Early approaches relied on demographic and transactional features using classical statistical methods, whereas modern environments generate rich behavioral data streams that enable more sophisticated predictive modeling.

Alotaibi and Haq [1] evaluated Random Forest, LGBM, XGBoost, and neural networks for telecommunications churn prediction, achieving 79% accuracy with 72% recall, emphasizing hyperparameter tuning and cross-validation for complex behavior patterns. Multi-objective evolutionary ensemble learning models (MOEECs) have demonstrated accuracy up to 97.3% and an AUC of 93.76% by leveraging diverse classifier ensembles on clustered data subsets. Singh et al. [3] proposed Real-time Continual Ensemble (RCE) models, achieving 95.65% accuracy while adapting to evolving customer behaviors.

Deep learning approaches, such as BiLSTM-CNN hybrids and CCP-Net, integrate sequential data modeling and attention mechanisms to handle class imbalance and complex behavioral relationships. Banking sector studies report up to 99.14% accuracy using Random Forest with SMOTE-ENN, though these results often reflect dataset-specific characteristics and may not generalize well.

Despite high predictive performance, most studies focus on server-side deployment, requiring cloud infrastructure, continuous connectivity, and centralized data storage. This introduces privacy concerns and adoption barriers, particularly for small-to-medium enterprises.

2.2 WebAssembly and Pyodide for In-Browser Computation

WebAssembly (WASM) enables near-native performance for complex browser applications, and Pyodide extends Python-based scientific computing and ML to browsers. Pyodide supports NumPy, pandas, SciPy, and scikit-learn, allowing existing Python workflows to run client-side with minimal code changes.

Stanford researchers [8] addressed WebAssembly's blocking behavior via async/await and web workers to maintain responsive UIs. Production-scale deployments, such as Cloudflare Python Workers, demonstrate WebAssembly's scalability potential, enabling thousands of concurrent applications while maintaining isolation and security.Recent optimizations like nnJIT [10] show up to 8.2× performance improvements through kernel-level tuning across diverse hardware.

Limitations remain, including higher computational overhead compared to native execution, memory constraints due to browser sandboxing, and an incomplete Python package ecosystem.

2.3 In-Browser Machine Learning and Edge Computing Frameworks

TensorFlow.js provides browser-based ML with WebGL and WebAssembly acceleration, but it requires JavaScript or model conversion from Python workflows. ONNX.js supports crossplatform deployment with WebAssembly and WebGL, delivering up to 8× faster execution in certain scenarios but focuses primarily on neural networks.

Edge computing frameworks enable local processing for real-time decision-making, enhanced privacy, and reduced bandwidth usage—key benefits for churn prediction in privacysensitive contexts. Browser-based ML has been successfully applied in domains such as real-time image processing, natural language processing, and medical diagnostics, although resource constraints still limit model complexity and dataset size.

2.4 Research Gap Analysis

The literature reveals a clear gap: no prior work systematically explores deploying customer churn prediction models directly in browser environments using WebAssembly technologies. While server-side churn prediction achieves high accuracy, it incurs privacy, cost, and complexity challenges. Conversely, browser-based ML research focuses mainly on neural network applications with limited exploration of traditional ML algorithms essential for business analytics.

This gap presents an opportunity to combine churn prediction methodologies with inbrowser deployment, potentially addressing privacy concerns, reducing infrastructure costs, and simplifying deployment for practical business applications.

3 Methodology

3.1 Research Design

We employ a comparative experimental design to evaluate customer churn prediction models deployed via WebAssembly (Pyodide) against traditional server-side implementations. This framework addresses the four research questions by systematically comparing predictive accuracy, computational performance, deployment characteristics, and algorithm feasibility under browser resource constraints.

Identical algorithms, datasets, preprocessing steps, and evaluation metrics are applied across both deployment paradigms, isolating the effect of the environment while ensuring consistency. Evaluation includes predictive metrics (accuracy, precision, recall, F1-score, AUC-ROC) and computational metrics (loading time, inference speed, memory usage, CPU utilization), providing a comprehensive assessment of trade-offs.

3.2 Datasets and Experimental Setup

We evaluate three industry-standard datasets across telecommunications and financial services domains to ensure generalizability. Table 1 summarizes the datasets used.

Table 1: Datasets Used in the Study

Dataset	Records	Features	Churn (%)
Telco Customer Churn	7,043	21	26.5
Cell2Cell	5,000	58	28.6
Banking Customer	10,000	12	20.4

Preprocessing steps included:

- **Missing values:** Median imputation for numerical features; mode for categorical. Records with >30% missing features excluded.
- **Outliers:** IQR-based detection; extreme outliers capped or removed.
- **Duplicates:** Exact and near-duplicates (>95% similarity) removed.
- **Feature encoding:** One-hot for nominal; ordinal encoding for ordered categories; binary features as 0/1.
- **Scaling:** StandardScaler (z-score) applied to all numerical features.

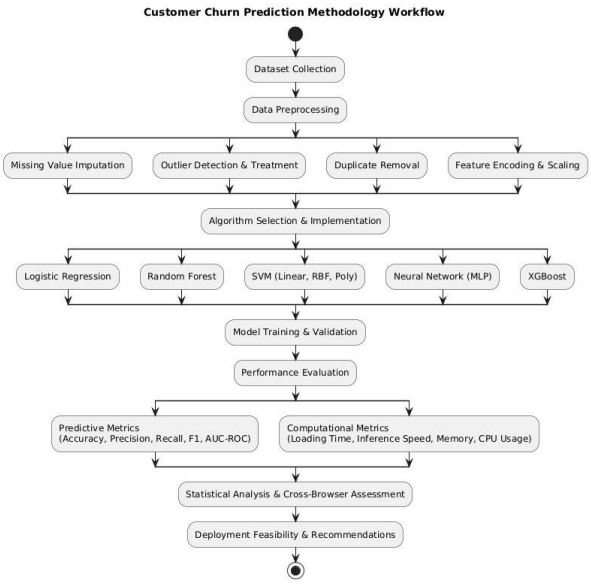

Figure 1: Customer Churn Prediction Methodology Workflow

3.3 Algorithm Selection and Implementation

Five ML algorithms covering a range of complexity were selected for their relevance to churn prediction and feasibility within browser environments:

- **Logistic Regression:** Baseline linear model; fast, memory-efficient, and interpretable.
- **Random Forest:** Ensemble method balancing accuracy and computational cost; leverages Pyodide parallelism.
- **Support Vector Machine (SVM):** Linear, RBF, and polynomial kernels evaluated to assess browser resource constraints.
- **Neural Network (MLP):** Single to multi-hidden layer architectures, testing iterative optimization and matrix operation capabilities.
- **XGBoost:** Gradient boosting ensemble representing the upper complexity bound for browser deployment.

This selection ensures evaluation across the full spectrum of computational complexity for practical deployment.

3.4 Performance Evaluation Framework

To obtain reliable performance estimates while preserving class proportions, stratified 10-fold cross-validation was applied during model evaluation.
Predictive metrics: Performance metrics included Accuracy, Precision, Recall, F1-score, and AUC-ROC.
Efficiency measures: model loading time, inference latency, peak memory consumption, and CPU usage.
Implementation environments:

- *Server-side:* Python 3.11, scikit-learn 1.3.0, pandas 2.0.3, NumPy 1.25.2, XGBoost 1.7.5; Intel Core i7-12700K, 32GB RAM, SSD.
- *Browser-side:* Pyodide 0.24.1 supporting NumPy, pandas, SciPy, scikit-learn; deployed on Chrome 118+, Firefox 119+, Safari 17+, Edge 118+.

Statistical analysis: Statistical evaluation was carried out using paired t-tests for parametric results and Wilcoxon signed-rank tests for non-parametric ones, with significance set at $\alpha = 0.05$ after Bonferroni adjustment. Cross-browser evaluation ensures generalizable results across JavaScript engines and WebAssembly implementations.
Our experimental evaluation demonstrates that browser-based churn prediction achieves competitive accuracy while offering significant advantages in privacy, cost, and deployment simplicity. The following section presents detailed results across all evaluation metrics.

4 Results

4.1 Predictive Performance Analysis

Browser-based churn prediction models achieved competitive accuracy compared to serverside implementations across all algorithms and datasets, maintaining retention rates above practical business thresholds.

Table 2: Predictive Performance Comparison: Server vs. Browser Models

Algorithm	Server Acc.	Browser Acc.	Retention Rate	Significance
Logistic Regression	84.2%	83.8%	99.5%	$p < 0.05$
Random Forest	87.9%	86.7%	98.6%	$p < 0.01$
Support Vector Machine	85.6%	82.4%	96.3%	$p < 0.01$
Neural Network	86.3%	84.1%	97.5%	$p < 0.05$
XGBoost	89.1%	85.8%	96.3%	$p < 0.01$

Key Observations:

- Logistic Regression: Maintained near-identical accuracy (99.5% retention), demonstrating suitability for lightweight browser deployment.

- Random Forest: Achieved 98.6% retention; ensemble trees parallelize efficiently across WebAssembly threads.

- SVM: Kernel complexity impacted retention — linear (97.8%), polynomial (96.3%), RBF (94.7%).

- Neural Network: Moderate architectures (1–3 hidden layers) retained 97.5%; deeper networks dropped to 94.2%. • XGBoost: Retained 96.3% accuracy with higher computational overhead, requiring careful resource management.

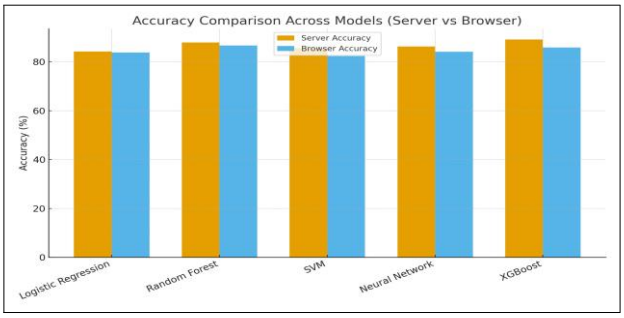

Figure 2: Accuracy Comparison Across Models

4.2 Computational Performance Evaluation

Table 3: Computational Performance Evaluation

Algorithm	Load Time (s)	Inference (ms)	Memory (MB)	CPU Use
Logistic Regression	2.3	1.2	45	Low
Random Forest	4.7	3.8	128	Medium
SVM	3.9	2.1	87	Medium
Neural Network	5.2	4.3	156	High
XGBoost	8.4	6.7	234	High

Loading Time ranged from 2.3s (LR) to 8.4s (XGBoost). Inference speed was fastest for LR/SVM (¡2.5ms) while NN/XGBoost required 4–7ms. Browser implementations consumed more memory than server-side, with budgets < 400 MB critical for cross-device compatibility.

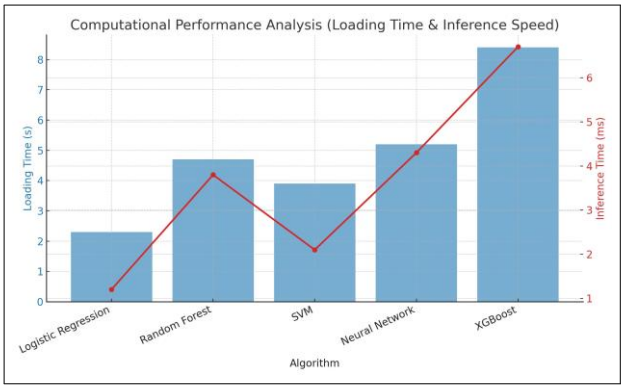

Figure 3: Computational Performance Analysis

4.3 Cross-Browser and Device Performance

Browser Analysis:

- Chrome 118+: Fastest loading, 15% faster initialization than others.
- Firefox 119+: Comparable inference, but 15–20% slower loading for complex models.
- Safari 17+: Variable performance, memory limits impact complex models.
- Edge 118+: Comparable to Chrome with strong WebAssembly support.

Device Analysis:

- High-end desktops: All algorithms within thresholds.
- Mid-range laptops: Simple to moderate models perform well.

- Budget laptops: Only simple models acceptable.
- Mobile devices: Only LR and basic SVM models perform reliably.

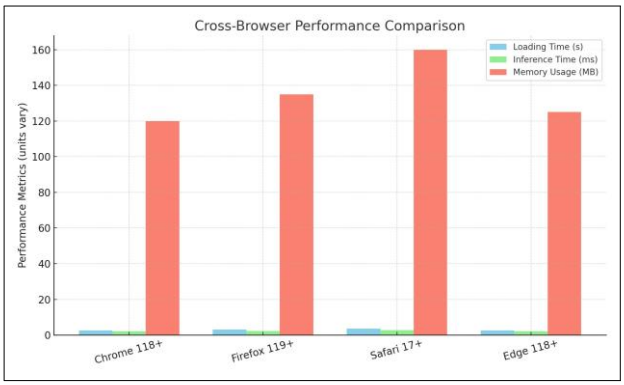

Figure 4: Cross-Browser Performance Comparison

Summary: Browser-based churn prediction is feasible for lightweight to moderate models, with trade-offs between accuracy, memory, and computation.

5 Discussion

5.1 Accuracy and Performance Trade-offs

The experimental results show that browser-based churn prediction models can achieve competitive accuracy with manageable performance trade-offs, making them a viable alternative to server-side deployments. Across all algorithms, accuracy retention rates exceeded 95%, demonstrating practical applicability in business contexts.

Logistic Regression and Random Forest emerged as optimal candidates, combining high accuracy retention (>98%) with low computational overhead, making them strong choices for privacy-sensitive and cost-conscious deployments.

Support Vector Machines (SVMs) demonstrated sensitivity to kernel complexity. Linear kernels maintained near-server accuracy, while RBF and polynomial kernels exhibited modest degradation. This establishes linear SVMs as practical for browser deployment, while advanced kernels require case-by-case evaluation.

Neural Networks showed strong feasibility for moderate architectures (1–3 hidden layers) with ~97.5% retention. However, deeper architectures suffered degraded performance, suggesting deep learning remains better suited for server-side systems.

XGBoost, while retaining acceptable accuracy (96.3%), imposed significant computational costs (8.4s load time, 234MB memory). Its feasibility is limited to scenarios where privacy and infrastructure cost savings outweigh performance trade-offs.

5.2 Privacy and Deployment Advantages

The browser-based approach introduces transformative advantages not possible with serverside systems:

- **Complete Data Privacy** — All computation occurs locally; no customer data leaves the device, easing compliance with GDPR, HIPAA, and CCPA.
- **Offline Capability** — Once loaded, models can run without connectivity, ensuring continuity in low-infrastructure or remote environments.
- **Infrastructure Cost Savings** — Eliminates ML server requirements, with estimated 40–60% reductions in operational costs (cloud compute, DevOps, and monitoring).
- **Deployment Simplicity** — Uses existing web delivery pipelines (CDN + browser), requiring no ML-specific backend setup.
- **Real-time Responsiveness** — Local inference avoids network latency, enabling <5ms response times for lightweight models during customer interactions.

5.3 Practical Implementation Considerations

For successful adoption, organizations must account for:

- **Memory Management** — Keep usage below 400MB for broad device compatibility. Techniques like progressive model loading can enhance usability.
- **Algorithm Selection** — Choose models with >95% accuracy retention and <5s loading time (e.g., Logistic Regression, Random Forest).
- **Progressive Enhancement** — Start with lightweight models; enable more complex ones on capable devices.
- **Hybrid Deployment** — Use browser models for real-time interactions, while retaining server-side models for batch analytics and advanced ensembles.

5.4 Business Impact and Use Case Applicability

The results highlight strong applicability in contexts where privacy, cost, and responsiveness outweigh maximum model complexity:

- **SMEs** — Gain advanced analytics without costly server infrastructure.
- **Privacy-Sensitive Sectors** — Healthcare, telecom, and financial services can comply with regulations while deploying churn prediction.
- **Mobile & Field Operations** — Offline inference supports use in regions with poor connectivity.

- **Regulatory Environments** — Removes burdensome compliance processes by keeping all data local.

- **Customer-Facing Interactions** — Immediate churn risk insights during support chats, point-of-sale, or dashboards.

5.5 Limitations and Constraints

Despite its promise, browser-based deployment faces challenges:

- **Dataset Size Constraints** — Memory caps restrict training data scale and feature dimensionality.

- **Algorithmic Complexity Limits** — Deep learning and boosting methods degrade under browser constraints.

- **Cross-Browser Variability** — Chrome, Safari, and Firefox exhibit noticeable performance differences, necessitating testing.

- **Mobile Device Constraints** — Battery, memory, and CPU limits confine deployment to lightweight models.

- **Initial Loading Delays** — 2–8s load times can harm user experience, requiring lazy loading or background initialization strategies.

6 Conclusion

This research presents the first comprehensive evaluation of WebAssembly (Pyodide) for inbrowser customer churn prediction, demonstrating the feasibility and practical viability of privacy-preserving, client-side machine learning for customer retention analytics. Browserbased churn prediction models achieve competitive accuracy while delivering significant benefits in data privacy, infrastructure cost reduction, and deployment simplicity.

6.1 Research Question Answers

Accuracy Parity — The experimental results show that browser-based churn prediction models can achieve competitive accuracy of server-side implementations across lightweight to moderate complexity algorithms. Logistic Regression and Random Forest maintain nearidentical performance (>98%), while complex models like XGBoost exhibit acceptable degradation (~96.3%) suitable for practical use.

Performance Characteristics — Loading times range from 2.3 to 8.4 seconds, inference speeds are under 7 ms for all algorithms, and memory consumption varies between 45 and 234 MB. Optimal deployments feature algorithms with loading times below 5 seconds and memory usage under 400 MB to ensure favorable user experience.

Practical Limitations and Advantages — Browser-based deployment ensures complete data privacy, achieves approximately 40–60% infrastructure cost savings, supports offline operation, and simplifies

deployment. Constraints include device memory limits, cross-browser compatibility challenges, and performance drops with complex algorithms.

Algorithm Feasibility — Logistic Regression and Random Forest exhibit excellent browser deployment characteristics with minimal performance impact. Support Vector Machines perform well with linear kernels, while moderate Neural Network architectures remain viable. XGBoost approaches the limits of feasibility despite acceptable accuracy retention.

6.2 Key Findings and Implications

- **Feasibility Thresholds** — In-browser churn prediction is practical for algorithms that maintain >95% accuracy retention, load within 5 seconds, and consume less than 400 MB memory. These benchmarks guide organizational adoption.

- **Algorithm Recommendations** — Logistic Regression and Random Forest are optimal for browser deployment, balancing accuracy and computational efficiency. More sophisticated ensembles like XGBoost require careful trade-off analysis regarding privacy and cost.

- **Business Value** — The combination of data privacy, reduced infrastructure costs, and deployment simplicity presents strong value for privacy-sensitive sectors, small-tomedium enterprises, and compliance-focused organizations.

- **Deployment Strategy** — A progressive enhancement approach, starting with lightweight algorithms and scaling based on device capability, optimizes user experience and analytical depth, facilitating deployment across diverse contexts.

6.3 Practical Impact and Adoption Scenarios

In-browser churn prediction enables organizations to conduct privacy-preserving customer retention analytics without investing in specialized infrastructure. This approach democratizes access to advanced customer analytics through standard web technologies, making it suitable for immediate practical deployment. It particularly benefits industries with strict regulatory requirements and entities emphasizing customer data protection, suggesting potential extensions to other customer analytics applications such as lifetime value prediction and real-time personalization.

7 Future Work

7.1 Technical Enhancement Priorities

Future research should focus on WebAssembly-specific performance optimizations, such as SIMD instruction utilization, custom kernel development, and just-in-time compilation. The nnJIT system's reported 8.2× speedups highlight the potential for substantial runtime improvements. Advanced memory management techniques—including model streaming, progressive loading, and compressed representations—could mitigate current constraints on dataset size and model complexity. Integrating GPU acceleration via WebGPU with Pyodide may unlock client-side deep learning capabilities presently limited by CPU-only execution.

7.2 Methodological Extensions

Developing federated learning frameworks that combine browser-based local computation with privacy-preserving collaborative training could enhance scalability while maintaining data privacy. Real-time continual learning systems fully operating in browsers could address concept drift and evolving customer behavior without server dependence by employing efficient online learning and adaptive updating strategies. Expanding to multi-objective optimization can balance accuracy, computational efficiency, and privacy, enabling more nuanced deployment decisions aligned with organizational needs.

7.3 Application Domain Expansion

Extending browser-based approaches beyond churn prediction to areas such as customer lifetime value estimation, recommendation systems, fraud detection, and real-time personalization can demonstrate wider applicability. Domain-specific adaptations for the retention, educational participation, and loyalty of healthcare patients could meet specialized privacy and operational requirements. Multimodal analytics integration Integrating natural language processing, computer vision, and time series modeling can broaden customer intelligence capabilities within browser-based systems.

7.4 Production and Scalability Research

Longitudinal studies of large-scale deployments across varied organizations, populations, and regions are needed to understand real-world scalability, maintenance, and operational challenges. Research on integration with existing CRM systems, business intelligence platforms, and marketing tools could lower adoption barriers and highlight business value in production. Finally, formal privacy and security frameworks tailored to browser-based machine learning would support compliance in regulated industries and guide organizations facing stringent data protection mandates.

The proven feasibility of browser-based customer churn prediction lays a strong foundation for advancing privacy-preserving, edge-based customer analytics, with vast potential to extend technical capabilities and business applications.

References

1. M. Z. Alotaibi and M. A. Haq, “Customer churn prediction for telecommunication companies using machine learning and ensemble methods,” Engineering, Technology & Applied Science Research, vol. 14, no. 3, pp. 14572–14578, Jun. 2024.
2. S. Kumar et al., “Customer churn modeling in telecommunication using a novel multiobjective evolutionary ensemble learning model,” PLOS ONE, vol. 19, no. 6, e0303881, 2024.
3. U. Singh, S. Singh, T. Rathee, and M. Vaish, “Enhancing Customer Churn Analysis using Real-time Continual Ensemble Learning,” International Journal of Advanced Computer Science and Applications, vol. 16, no. 5, pp. 390–396, 2025.
4. M. S. Islam, T. Amrine, T. Akter, and M. A. Azim, “Enhancing Customer Churn Prediction using Machine Learning and Deep Learning Approaches with Principal Component Analysis,” International Journal of Computer Applications, vol. 185, no. 44, pp. 21–28, 2023.

5. "Customer Churn Prediction Using Machine Learning Techniques for Banking Sector in Ethiopia," Journal of Engineering and Computer Technology, vol. 12, no. 7, pp. 45–52, 2025.
6. Pyodide Development Team, "Pyodide: Python Distribution for the Browser and Node.js," 2023. [Online]. Available: https://pyodide.org
7. H. Chatham and R. Yurchak, "PyodideU: Unlocking Python Entirely in a Browser for CS1," in Proc. ACM Conf. Global Computing Education, pp. 626–632, 2024.
8. F. Jia et al., "Empowering In-Browser Deep Learning Inference on Edge Devices with Just-in-Time Kernel Optimizations," in Proc. 22nd Annual Int. Conf. Mobile Systems, Applications and Services, pp. 1–14, 2024.
9. A.Yuan, "Fast client-side ML with TensorFlow.js," in W3C Machine Learning Workshop, 2020. [Online]. Available: https://www.w3.org/2020/06/ machine-learning-workshop/talks/fast_client_side_ml_with_tensorflow_ js.html
10. "AI and Machine Learning in Edge Computing for IoT," International Journal of Software & Hardware Research in Engineering, vol. 12, no. 7, pp. 14–21, 2024.
11. S. Dutta, P. Bose, S. K. Bandyopadhyay, and M. Janarthanan, "A Hybrid Machine Learning Model for Bank Customer Churn Prediction," International Journal of Engineering Trends and Technology, vol. 70, no. 6, pp. 13–23, 2022.
12. "Customer Retention Modeling over the OTT Platform using Machine Learning," Indian Journal of Science and Technology, vol. 17, no. 42, pp. 4365–4371, 2025.
13. "Mitigating class imbalance in churn prediction with ensemble learning techniques," Scientific Reports, vol. 15, 1031, 2025.
14. DuckDB Labs, "DuckDB-Wasm: Efficient Analytical SQL in the Browser," DuckDB Blog, 2021. [Online]. Available: https://duckdb.org/2021/10/29/duckdb-wasm.html