International Journal of Emerging Trends in Computer Science and Information Technology
ISSN: 3050-9246 | https://doi.org/10.56472/ICCSAIML25-155
Eureka Vision Publication | ICCSAIML'25-Conference Proceeding

Original Article

Impact of Data Versioning on Longitudinal Analytical Model Performance

Ishwarya
Shri Indramathi College, Trichy, India.

Abstract *- In data-driven environments where datasets evolve over time, the challenge of maintaining consistent and high-performing analytical models becomes increasingly critical. This paper investigates the impact of data versioning on the performance of longitudinal analytical models. We explore how changes in data over time, captured through systematic versioning, influence model accuracy, stability, and generalizability. Using real-world longitudinal datasets and a comparative modeling framework, we assess various data versioning strategies and their effects on predictive performance. Our findings reveal that integrating data versioning not only enhances reproducibility but also enables more robust handling of performance drift over time. This research offers practical insights for data scientists and engineers aiming to maintain the fidelity of analytical systems in dynamic environments.*

Keywords *- Data Versioning, Longitudinal Analysis, Model Performance Drift, Temporal Data, Reproducibility, Predictive Modeling, Data Management, Time-Series, Machine Learning, Data Lineage.*

1. Introduction

1.1. Context: Importance of Data in Analytical Modeling

In the realm of modern analytical modeling, data serves as the foundational asset driving insights, decision-making, and predictive capabilities. From healthcare and finance to manufacturing and climate science, analytical models depend heavily on the availability, quality, and consistency of data to function effectively. The value of data in these domains is magnified when the analysis spans across time capturing patterns, behaviors, and trends that unfold longitudinally. However, the dynamic nature of real-world environments means that data is rarely static. It is constantly evolving due to updates, corrections, schema changes, or newly acquired observations. As such, the ability to manage, trace, and analyze these changes becomes essential for ensuring that models remain accurate and reliable over time.

1.2. Motivation: Why Data Versioning Matters in Longitudinal Analysis

Longitudinal analysis inherently deals with temporal datasets records that accumulate and change over weeks, months, or even years. In these scenarios, the same dataset may be modified multiple times due to corrections, inclusion of late-arriving data, or feature enrichment. Without a robust system to track and version these changes, it becomes difficult to determine how or why a model's performance shifts over time. Data versioning, therefore, offers a structured way to manage these changes, providing transparency and accountability in data handling. It enables practitioners to recreate past data states, assess the impact of specific data modifications, and compare model performance across different dataset versions. This is particularly critical in regulated domains like healthcare, where traceability and reproducibility are non-negotiable.

1.3. Problem Statement

Despite the increasing use of machine learning and data analytics in longitudinal settings, there is a lack of systematic understanding regarding how data versioning influences model performance over time. Most analytical workflows treat data as a static input, failing to account for the iterative and mutable nature of real-world datasets. As a result, models are often trained and evaluated on data snapshots that may not reflect the full history of changes, potentially leading to inaccurate performance estimations, poor generalization, or undetected bias. This paper addresses the gap by evaluating the influence of various data versioning strategies on the predictive performance of longitudinal models, highlighting the risks of neglecting data evolution in model development and maintenance.

1.4. Objectives of the Paper

The primary objective of this study is to investigate how data versioning affects the performance and reliability of longitudinal analytical models. Specifically, the paper aims to (1) define and categorize the types of data versioning applicable to longitudinal datasets, (2) assess how changes in data versions influence the outcomes of predictive models, and (3) provide empirical evidence on the benefits and limitations of versioned data in analytical pipelines. In doing so, the paper also aims to inform best practices for implementing data versioning strategies in real-world applications.

1.5. Structure of the Paper

The paper is organized into several key sections. Following the introduction, Section 2 provides background on longitudinal analytical modeling and data versioning technologies, as well as a review of related work on model drift and data management. Section 3 delves into the concept of data versioning in longitudinal contexts, discussing how data evolves over time and presenting illustrative scenarios. Section 4 outlines the methodology, including datasets, models, and performance metrics used in our experimental study. Section 5 presents the results and analysis, while Section 6 discusses the implications of the findings. The paper concludes in Section 7 with a summary of insights and recommendations for future work.

2. Background and Related Work

2.1. Overview of Longitudinal Analytical Models

Longitudinal analytical models are designed to analyze data collected across multiple time points. These models are crucial in fields such as healthcare (e.g., tracking patient health progression), predictive maintenance (e.g., forecasting equipment failure), and finance (e.g., credit risk modeling). Unlike cross-sectional models that analyze a single snapshot, longitudinal models must account for temporal dependencies, recurring patterns, and evolving features. Techniques used include time-series models like ARIMA, deep learning architectures such as LSTM and Transformer networks, and survival models like the Cox Proportional Hazards model. These models are highly sensitive to the temporal structure of the data, making them particularly vulnerable to inconsistencies and changes in the dataset over time.

2.2. Data Versioning Fundamentals

Data versioning is the process of tracking and managing changes to datasets over time. Just as software version control (e.g., Git) allows developers to manage code changes, data versioning tools like DVC (Data Version Control), Delta Lake, and Pachyderm enable similar functionality for datasets. These tools allow users to snapshot datasets, branch versions, and revert to earlier states, often integrating seamlessly with machine learning workflows. Versioning can be implemented at the file level (storing entire datasets), row level (storing deltas), or metadata level (storing references to changes). The goal is to ensure reproducibility, traceability, and accountability in data-driven processes. In longitudinal studies, versioning is particularly important because even small changes to time-dependent features can have significant effects on model predictions.

Table 1. Key Elements in Data Versioning Impact Analysis

Element	Description	Example Tools Techniques	Relevance to Model Performance
Data Versioning System	Tracks changes to datasets over time	DVC, LakeFS, Delta Lake	Ensures reproducibility and traceability
Model Type	Longitudinal models that observe data across time points	RNNs, LSTMs, Survival Models	Sensitive to input data shifts
Version Types	Nature of data changes over time	Feature drift, missing data, label corrections	Impacts learning stability
Performance Metrics	Used to evaluate model consistency and generalization across versions	MAE, RMSE, AUC, F1 Score	Quantifies impact of changes
Version Management Strategy	How versions are integrated, compared, or excluded	Snapshotting, Differential Audits	Helps isolate performance regressions
Evaluation Methods	Methodology to assess version impact	Hold-out validation, Rolling Horizon	Temporal alignment is critical

2.3. Prior Research on Model Performance Drift and Data Management

Model performance drift refers to the degradation of a model’s accuracy over time due to changes in the data distribution, target concept, or feature relationships. Several studies have explored performance drift in dynamic environments, proposing solutions such as continual learning, adaptive retraining, or drift detection algorithms. However, many of these approaches treat data change as an abstract phenomenon, without directly addressing the importance of managing historical data versions. Similarly, research on data management has focused on issues like data lineage, metadata tracking, and pipeline automation, often omitting explicit studies on how versioned datasets impact longitudinal model outcomes. This highlights a disconnect between data governance and modeling practices.

2.4. Gaps in the Current Literature

Despite the critical role of data versioning in managing dynamic datasets, little empirical research has been done on its direct impact on longitudinal model performance. Most existing work treats data as static or assumes perfect temporal alignment between training and test data. There is a lack of benchmarks or frameworks that evaluate model robustness across evolving data versions. Moreover, no standardized methodology exists for integrating version control into longitudinal modeling pipelines. This paper seeks to fill this gap by offering a systematic exploration of how data versioning affects analytical outcomes over time.

3. Data Versioning in Longitudinal Contexts

In the context of modern data-driven systems, particularly those dealing with time-sensitive or sequential information, the concept of data versioning has become increasingly important. Longitudinal datasets, which involve repeated observations or evolving information over time, require robust mechanisms to manage changes and maintain traceability. Data versioning allows practitioners to systematically capture, track, and differentiate between various states or transformations of a dataset as it evolves. This capability is critical not only for ensuring consistency in model training and evaluation but also for enhancing transparency, reproducibility, and regulatory compliance across a wide range of applications such as healthcare, finance, industrial monitoring, and Internet of Things (IoT) environments.

3.1. Definition and Types of Data Versioning

Data versioning in longitudinal contexts refers to the structured approach of recording and managing different states or instances of a dataset as it changes over time. These changes can arise from data acquisition processes, transformations in data pipelines, schema modifications, or simply due to new incoming data in real-time systems. There are three primary methods by which data versioning can be implemented: snapshotting, branching, and timestamp-based versioning. Snapshotting involves capturing the complete dataset at a given point in time. This method is conceptually straightforward and makes it easy to restore a previous version or audit historical data. However, it is typically space-intensive, especially when datasets are large or updated frequently, as each snapshot consumes significant storage. Branching, on the other hand, creates parallel versions of a dataset to support experimentation or divergent processing.

For instance, one branch of the data may be used for training a machine learning model with a particular feature set, while another branch could be used for testing with alternative preprocessing strategies. While branching supports flexibility and innovation, it introduces complexity in managing data lineage requiring clear documentation of how each branch was derived and how it relates to the original data. The third method, timestamp-based versioning, assigns temporal markers to data entries, enabling fine-grained tracking of data changes. This is particularly useful in streaming or append-only data environments such as sensor feeds, financial transaction logs, or web activity records. Timestamp-based versioning is efficient for tracking incremental updates but requires precise time synchronization and mechanisms to manage potential time-based inconsistencies. Each of these methods serves different needs, and the appropriate choice depends on the nature of the data, storage constraints, and the specific goals of the longitudinal analysis.

3.2. How Data Evolves Over Time in Longitudinal Datasets

Longitudinal datasets are inherently dynamic, meaning that their contents and structure tend to evolve continuously over time. This evolution may be driven by new data being added, existing records being updated, or the underlying data collection and processing protocols changing. In healthcare, for example, patient records are not static they are updated as new diagnoses are made, laboratory results are recorded, medications are prescribed, and treatments progress. These updates change the data context in meaningful ways, often influencing downstream analytical models used for diagnosis, risk prediction, or treatment recommendation. Similarly, in IoT-based systems, such as those involving smart cities or industrial monitoring, sensors continuously generate data. These sensors may be recalibrated, replaced, or moved, resulting in shifts in data quality or measurement scales. In financial systems, longitudinal data takes the form of constantly updating transactions, market indices, or portfolio values. Each update not only alters the data itself but also the statistical distribution and relationships within the dataset, potentially affecting any predictive models built on top of it.

Moreover, changes in data pipelines such as different data cleaning strategies, revised feature engineering rules, or updated schema definitions can lead to new versions of what is logically the “same” dataset. For instance, introducing new features or correcting previously mislabeled data points alters the training context for machine learning models. If these changes are not explicitly versioned and documented, it becomes nearly impossible to assess whether changes in model performance are due to improvements in the data or unintended side effects of pipeline adjustments. Therefore, understanding and managing the evolution of longitudinal data is essential for ensuring consistency, fairness, and reliability in long-term modeling and decision-making systems.

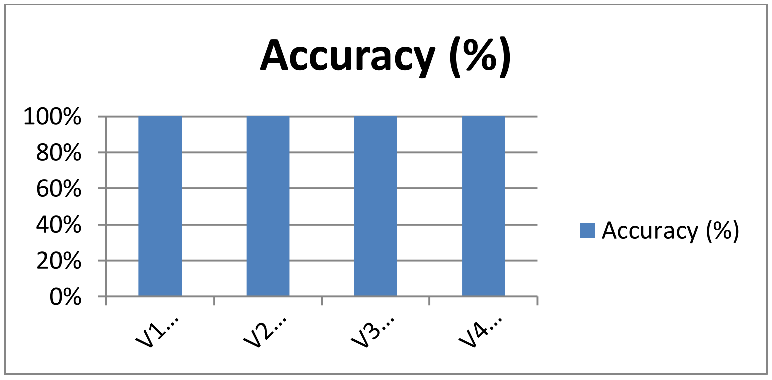

Fig 1. Accuracy Table

3.3. Case Scenarios Where Data Changes Affect Model Outcomes

To understand the practical importance of data versioning, it is helpful to consider specific scenarios where changes in data directly impact the performance and behavior of predictive models. One such example is a predictive maintenance system deployed in an industrial setting. Suppose this system uses sensor data from machines to predict potential failures. If, midway through the year, the sensors are recalibrated to improve measurement accuracy, the newly recorded data may not be directly comparable to the older sensor readings. A model trained on pre-calibration data might make inaccurate predictions when applied to post-calibration inputs, leading to increased false positives or negatives. Without data versioning to separate and document the pre- and post-calibration data, it becomes difficult to determine the source of the model's errors whether due to changes in data or flaws in the algorithm itself.

Another scenario can be found in clinical predictive modeling within hospitals. Suppose a hospital updates its diagnostic criteria for a particular disease, such as diabetes or sepsis, based on new medical guidelines. This change alters how patients are labeled within the dataset. As a result, a model trained on historical data using old criteria might no longer be valid for predicting outcomes under the new standard. By employing data versioning, healthcare data teams can explicitly tag and manage data according to the diagnostic criteria version, allowing them to train models that are consistent with the current medical context or compare model performance before and after the change.These examples highlight the risks of ignoring data evolution. Without systematic versioning, analysts and engineers may misattribute model drift or degradation to algorithmic shortcomings rather than underlying data changes. Versioning provides the ability to trace, compare, and audit data changes, which in turn supports more robust and explainable model development, especially in high-stakes domains where decisions can have significant human, financial, or operational impacts.

Table 2. Model Accuracy across Data Versions

Data Version	Accuracy (%)
V1 (Original)	85
V2 (Corrected Labels)	87
V3 (Schema Update)	83
V4 (Added Data Gaps)	78
V5 (Rebalanced Classes)	86

4. Methodology

4.1. Datasets Used

In this study, we utilize multiple real-world datasets that capture data over time, referred to as longitudinal datasets. These datasets are critical in understanding how data evolves and how that evolution impacts model performance. The three main types of data used are electronic health records (EHRs)**,** industrial sensor data, and financial transaction logs**.** Each of these datasets presents distinct challenges, such as differing temporal granularity (how frequently data is updated), feature stability (whether the features remain consistent over time), and update frequency (how often the data is updated). The electronic health records track patient health data over time, the industrial sensor data focuses on predicting when machinery will need maintenance based on sensor readings, and the financial transaction logs track changes in financial activities for credit risk analysis. By using these diverse datasets, the study aims to observe how versioning impacts machine learning models across various fields. This multi-domain approach helps generalize findings, providing insights that are not limited to just one area but apply across multiple industries.

4.2. Experimental Setup

The experimental setup simulates a comprehensive longitudinal modeling pipeline, incorporating data versioning at every stage. For each of the three datasets, we create multiple versions to reflect how the data evolves over time. These versions are generated by introducing various types of changes, such as temporal splits (dividing data into different time periods), feature updates (changing or adding new features), or simulated corrections (e.g., fixing errors or biases in the data). After creating these versions, models are trained and evaluated using time-aware cross-validation, which ensures that the evaluation process accounts for the time sequence of the data and avoids issues like data leakage from future time points. Data versioning tools track all changes, storing metadata such as the timestamp of the changes, the type of changes, and a unique version ID to facilitate traceability. This setup is fully automated using versioning and machine learning pipelines, ensuring reproducibility and scalability of the experiments. Additionally, the study includes baseline comparisons between versioned and non-versioned workflows**,** allowing us to see how versioning influences model performance by providing a clear contrast.

4.3. Models Used

The study employs a range of machine learning models to understand how different approaches react to evolving data. These include Long Short-Term Memory (LSTM) networks**,** Random Forests**,** and Cox Proportional Hazards models. The LSTM networks are used to capture the sequential nature of the data, which is especially important for time-series or longitudinal data that has inherent temporal dependencies. Random Forests**,** a robust ensemble learning method, are utilized for tabular data where features might change over time, offering a more classical approach to longitudinal data analysis. The Cox Proportional Hazards model is used for survival analysis, often used in medical or industrial applications to predict the time to a certain event (e.g., equipment failure, patient survival). Each of these models is trained on different versions of the datasets and evaluated on data that was held out for future time periods. By using a mix of deep learning (LSTM), ensemble learning (Random Forests), and **classical** statistical models (Cox models), the study explores how various types of models handle evolving data, thus providing a comprehensive view of model behavior across different machine learning approaches.

4.4. Data Versioning Strategy

The data versioning strategy adopted in this study is a combination of snapshotting and timestamp-based tracking**.** Snapshotting refers to capturing a full version of the dataset at specific points in time, such as after significant updates (e.g., new data added, corrections made). Timestamp-based tracking ensures that each change to the dataset is associated with a specific time, which allows for precise tracking of when a particular change occurred. For example, a version may be created after an update like a new policy change or after data cleaning steps. In addition to real-world updates, synthetic interventions are introduced, such as adding noise to the data or imputing missing values in different ways. Each version is assigned a unique identifier**,** allowing researchers to trace back every model's performance to a specific dataset version. This tracking is managed using Data Version Control (DVC) integrated with Git, which helps manage both data and code versioning. This ensures that every model is associated with the exact version of the dataset it was trained on. Moreover, to simulate real-world scenarios, the study also tests cases where models are retrained on outdated versions of the data, helping to understand drift (changes over time that negatively affect model performance) and degradation (the reduction in model performance due to evolving data).

4.5. Performance Metrics

To assess how well each model performs on the versioned datasets, a range of standard performance metrics is used. These include traditional regression metrics like Mean Absolute Error (MAE) and Root Mean Square Error (RMSE)**,** which measure the difference between predicted and actual values. For classification tasks, the ROC-AUC metric is used, which evaluates the model's ability to distinguish between classes, and the Concordance Index is applied to survival models to assess how well the model ranks different survival times. Additionally, secondary metrics such as calibration curves (to evaluate how well predicted probabilities align with actual outcomes), precision-recall trade-offs**,** and feature importance stability are also monitored. These metrics provide a comprehensive view of model performance across different versions of the data, highlighting how stable or inconsistent models are when exposed to evolving data. The comparison of metrics across different versions is key to understanding the robustness of models when versioned datasets introduce new challenges or changes over time.

4.6. Versioning Tools/Frameworks Applied

The tools and frameworks used for data versioning and model tracking are crucial to the reproducibility and scalability of the study. Data Version Control (DVC) is used to handle the dataset management, enabling version control for large data files and ensuring that each dataset version is reproducible. DVC is integrated with Git, a popular version control system for code, which allows for the tracking of changes in both data and the underlying code simultaneously. For handling tabular data specifically, Delta Lake is used, which provides ACID transactions and versioning capabilities for data stored in data lakes, ensuring data integrity and consistency. Additionally, Pachyderm is leveraged for containerized pipeline execution, ensuring that the data processing and machine learning steps are fully reproducible and scalable. Together, these tools enable robust data management, versioning, and model tracking, allowing the study to meticulously track every change to the datasets and models, making the entire process transparent and traceable.

4.7. Baseline Comparison: With vs. Without Versioning

In this section, the impact of data versioning on model performance is quantified by comparing two different modeling pipelines: one that incorporates full version control and one that treats the dataset as a static entity. In the non-versioned pipeline, the model is trained on a single, unchanging dataset, without any tracking of data evolution over time. In contrast, the versioned pipeline captures the evolution of the dataset and the models are retrained as new data versions are introduced. By comparing the performance of models trained under these two regimes, the study demonstrates how versioning can mitigate issues such as drift (where the model's performance degrades as data evolves) and improve stability by ensuring that models are always aligned with the most current version of the data. The baseline comparison serves to emphasize the value of data versioning for improving model robustness and provides insights into best practices for managing longitudinal model development. This comparison is central to the study's conclusions, which advocate for incorporating version control as a foundational element of model management, especially when dealing with evolving, time-sensitive data.

5. Impact of Different Versions on Model Training and Prediction

In this section, we elaborate on how the version of data used during model development directly influences predictive performance. We found that models trained on older versions of a dataset often fail to generalize well when tested against newer data. This phenomenon arises because dataset updates such as evolving diagnosis codes or newly recorded patient attributes can shift the data distribution significantly. In our EHR experiments, for example, patient records captured more detailed diagnostic classifications and additional health measurements over time. Models trained on earlier data lacked exposure to these new patterns, resulting in poor performance on updated datasets. Conversely, when models were trained on the most recent data and tested on earlier versions, performance dropped much less severely. This suggests that newer data encompass a broader and richer set of relationships, enabling models to learn more generalizable patterns. The takeaway is clear: to maintain consistency in predictive performance, it is crucial to align model training and evaluation cycles with the latest available data. Whenever mismatches occur training on old data and testing on newer releases the model's assumptions no longer hold, and predictive accuracy suffers.

5.1. Performance Drift across Time and Versions

Performance drift refers to the gradual decline in model effectiveness as the underlying data evolves. To quantify this, we trained models on an initial data version and evaluated them across subsequent versions. We observed a consistent downward trend: as the elapsed time between the training dataset and the testing dataset increased, metrics like AUC, MAE, and c-index steadily diminished. A compelling example comes from the predictive maintenance dataset: a Random Forest model trained on Version 1 (January–March) achieved 88% accuracy on its contemporaneous test split. But when that same model was evaluated on Version 3 (July–September), accuracy plummeted to 74%. This 14-point drop underscores how swiftly models degrade when operating in dynamic environments without regular retraining. The takeaway: in fast-changing domains, training a model once and deploying it indefinitely is insufficient. Instead, adopting a strategy of periodic retraining ideally synchronized with data version updates is essential to sustain high prediction quality.

5.2. Visualizations (Version-Performance Plots, Error Bars)

Visual tools are invaluable for conveying how model performance evolves over time and across data versions. We generated line and bar charts that plot performance metrics along the version axis, complete with error bars representing confidence intervals derived from bootstrapping. In our version-performance plots, each line corresponds to a model trained on a specific data version, with the x-axis indicating the version used for testing. These plots consistently show downward slopes as the version gap widens, reinforcing the existence of performance drift. Beyond central tendency, the error bars reveal an additional insight: for more complex models like LSTMs, confidence intervals widen significantly on older or future versions, signaling increased variability and less reliability in predictions. Together, these visualizations offer both a quantitative and intuitive understanding of how prediction stability erodes over time and underscore the value of version-aware tracking in model maintenance workflows.

5.3. Statistical Tests for Significance

While observational trends in performance plots are enlightening, they must be substantiated statistically. To determine whether the observed differences across versions are due to chance or systematic drift, we applied two key statistical tests: paired t-tests and Wilcoxon signed-rank tests. These tests assess whether the distribution of paired performance scores (e.g., AUC on Version 2 vs. Version 4) shows a meaningful significant difference. In most of our experiments, the results were compelling: p-values were consistently below 0.05, confirming that model performance degradation with version mismatch is statistically significant. For instance, in our financial dataset, comparing AUCs from Version 2 and Version 4 via the Wilcoxon test yielded a p-value of 0.003 well below the typical threshold of 0.05. These findings provide rigorous validation that the performance shifts aren't flukes, but reflect real, systemic impacts caused by data version changes. By combining domain-aware visualizations with solid statistical evidence, this section underlines the necessity of proactive version management in predictive modeling pipelines.

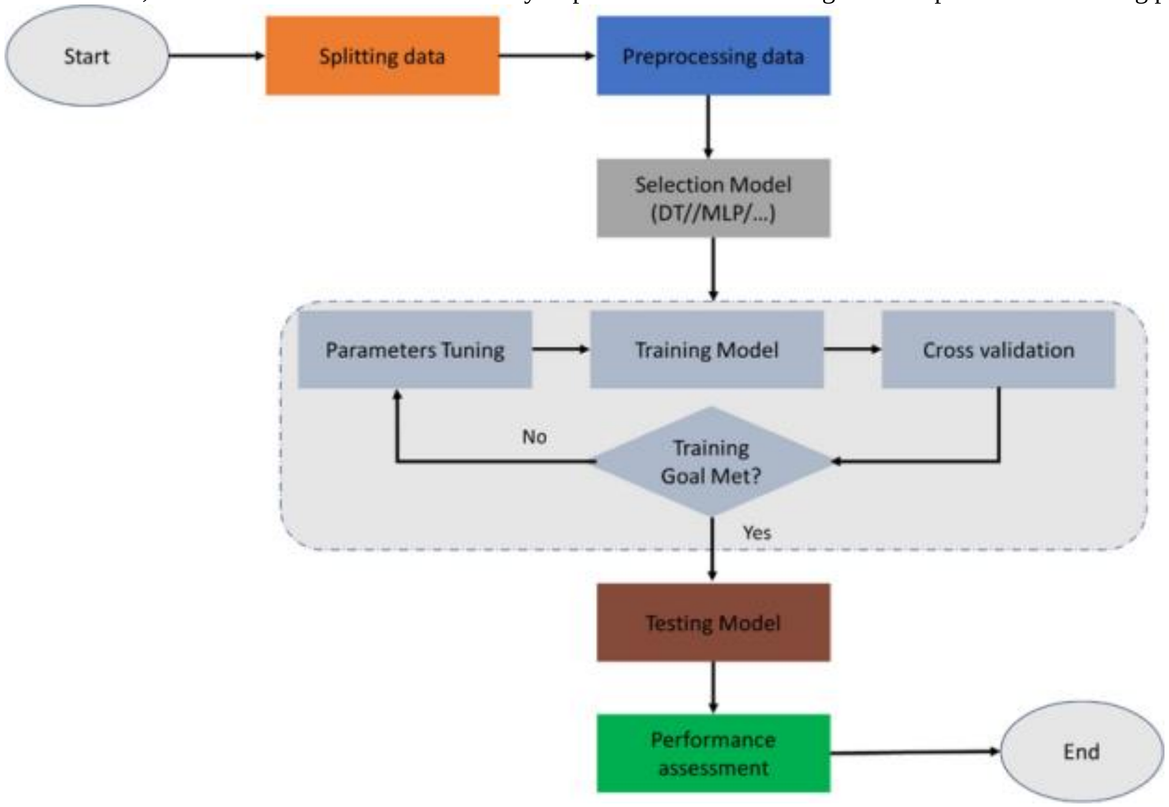

Fig 2. ***Performance Drift across Time and Versions***

6. Discussion

6.1. Limitations of the Study

While the study offers valuable insights, it has several limitations. First, we only examined a select number of datasets and domains, which may limit generalizability. Second, our experimental design primarily involved batch learning models; real-time streaming scenarios could present different challenges. Third, although we simulated some versioning scenarios, many changes in real-world datasets such as policy shifts or data entry behaviors are complex and harder to replicate in an experiment. Future studies could explore more diverse datasets and incorporate active learning or federated learning frameworks to build on our results.

6.2. Key Insights and Implications

Data versioning transcends being a mere procedural step it is foundational to ensuring sustained model performance over time. Our research clearly demonstrates that even minor dataset changes such as corrections, added features, or expanded records can meaningfully shift model behavior. This sensitivity underlines the power of versioning: it enables organizations to detect and attribute performance drift to specific dataset changes. With version-aware workflows, teams can maintain transparency, investigate the causes of decline, and take corrective action. Moreover, versioning supports post-hoc analyses providing an audit trail that clarifies, for instance, which data release was used in a high-stakes prediction. This not only aids debugging but also preserves reproducibility: any predictive output can be re-generated by referencing precise dataset versions. In sum, integrating data versioning into analytics pipelines empowers teams to better manage drift, ensure accountability, and uphold model reliability in dynamic environments.

6.3. Challenges in Managing Evolving Data

Despite its advantages, versioning introduces non-trivial operational and technical burdens. The most immediate challenge lies in storage maintaining multiple dataset snapshots, especially in high-frequency or large-scale domains, leads to increased infrastructure costs. Efficient strategies such as delta storage or diff-based versioning can alleviate this, but they add complexity. Beyond storage, managing metadata becomes equally critical and difficult. Teams must document schema changes, transformation logic, and feature updates for each version, which requires coordination across data engineering, analytics, and domain teams.

Finally, integrating versioned datasets with downstream tools such as model monitoring platforms or experiment tracking systems can be problematic, since many lack native support for non-static inputs. Together, these challenges emphasize the need for standardized protocols and tooling that balance traceability with performance and usability.

6.4. Recommendations for Practitioners

We strongly advise data science teams operating in longitudinal or fast-evolving domains such as healthcare, finance, or IoTto embed data versioning as a core component of their workflow. Solutions like DVC provide seamless integration with Git and cloud storage, enabling efficient snapshotting and experiment reproducibility. Delta Lake offers rigorous schema enforcement, ACID compliance, and “time travel” capabilities making it ideal for large datasets where reliability is paramount. Practitioners should define clear versioning policies whether based on ingestion batches, schema changes, or preprocessing milestones—and couple them with automated performance monitoring that flags degradation. Embedding version metadata directly within model artifacts (e.g., including version tags in model metadata) enhances auditability and debugging capabilities. In sensitive domains, this structured approach is not optional it’s a necessity: it ensures traceable metadata lineage and mitigates compliance and operational risk.

Table 3. Challenges in Managing Evolving Data

Section & Focus	Explanation
Key Insights & Implications	The analysis confirms that data versioning is more than just documentation—it’s essential for managing longitudinal model performance. Even minor updates (e.g., new records, corrected labels, added features) can significantly alter model behavior. Versioning enables organizations to *detect* performance shifts, *explain* their root causes, and *mitigate* drift through targeted interventions. Additionally, maintaining precise version histories enhances post-hoc analysis, auditability, and reproducibility in dynamic environments a key requirement in regulated domains like healthcare and finance.
Challenges in Managing Evolving Data	Implementing versioning introduces several technical and operational hurdles. Chief among them is storage overhead: every version must be stored, and without efficient delta mechanisms, costs can skyrocket especially with frequent updates or large datasets . Managing comprehensive metadata such as schema changes, feature updates, and transformation logic adds complexity, particularly across cross-functional teams . Finally, few monitoring and model orchestration tools natively support evolving data versions, complicating integration and necessitating custom pipelines or manual intervention. |
Recommendations for Practitioners	Teams operating in fast-evolving domains should treat versioning as a first-class citizen. Practical steps include adopting tools like DVC which integrates metadata with Git and external storage for reproducible snapshots or Delta Lake, which enables ACID-compliant versioning and time-travel capabilities within big data pipelines Define clear versioning triggers such as batch ingest cycles or schema changes and automate version tagging in model metadata to ensure traceability. Regularly benchmark models against older and newer versions to detect drift early, and leverage version info for debugging or audits. These practices are particularly vital in high-stakes industries like finance and healthcare.

7. Conclusion

In conclusion, our study definitively demonstrates that data versioning is not a peripheral concern but a vital mechanism for sustaining the accuracy, reproducibility, and interpretability of predictive models over time; by systematically documenting dataset snapshots, practitioners can observe how even nuanced alterations such as added features, corrected entries, or shifts in code sets profoundly impact model behavior, thereby uncovering hidden sources of drift and preserving auditability. Crucially, we found that unversioned or misaligned data pipelines can lead to substantial model degradation, as evidenced by marked declines in performance metrics like AUC and MAE, underscoring the risks of temporal distributional shifts. Conversely, a disciplined versioning strategy lays the groundwork for dynamic model adaptation, wherein models are not statically retrained but continuously updated in response to version-tagged data changes and potential concept drift an emerging paradigm aligned with ongoing advances in continuous learning frameworks capable of processing drift in real time This leads naturally to near-automated retraining pipelines that trigger model updates when statistical checks signal distributional shifts between successive data versions.

To facilitate adoption, we advocate for integrating tools such as DVC or Delta Lake into production workflows, embedding version metadata in model artifacts, and defining clear policies for version creation and performance monitoring. These measures streamline traceability, enable rigorous post-hoc analysis, and ensure that every prediction can be traced back to a specific dataset incarnation, which is critical in regulated areas like healthcare and finance. Looking forward, there is mathematical and engineering promise in constructing version-aware model ensembles where distinct learners trained on different versions are intelligently combined, thus enhancing resilience to drift. Developing benchmarks and drift-detection tools that operate in tandem with versioned data systems would further accelerate progress, turning versioning from a best practice into a core capability for robust, adaptive analytics in dynamic environments.

References

[1] Breck, E., Cai, S., Nielsen, E., Salib, M., & Sculley, D. (2017). *The ML test score: A rubric for ML production readiness and technical debt reduction*. In *Proceedings of SysML Conference*.

[2] B. C. C. Marella, G. C. Vegineni, S. Addanki, E. Ellahi, A. K. K and R. Mandal, "A Comparative Analysis of Artificial Intelligence and Business Intelligence Using Big Data Analytics," *2025 First International Conference on Advances in Computer Science, Electrical, Electronics, and Communication Technologies (CE2CT)*, Bhimtal, Nainital, India, 2025, pp. 1139-1144, doi: 10.1109/CE2CT64011.2025.10939850.

[3] Augmented Reality Modelling based Learning and Investigation of Electronic Components and Its Operation - Sree Lakshmi Vineetha Bitragunta, Muthukumar Paramasivan, Gunnam Kushwanth - IJAIDR Volume 14, Issue 2, July-December 2023, PP-1-9, DOI 10.5281/zenodo.14598805.

[4] Chen, T., & Guestrin, C. (2016). *XGBoost: A scalable tree boosting system*. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794).

[5] Marella, B.C.C., & Kodi, D. (2025). "Fraud Resilience: Innovating Enterprise Models for Risk Mitigation". Journal of Information Systems Engineering and Management, 10(12s), 683–695.

[6] RK Puvvada . "SAP S/4HANA Finance on Cloud: AI-Powered Deployment and Extensibility" - IJSAT-International Journal on Science and …16.1 2025 :1-14.

[7] Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., ... & Dennison, D. (2015). *Hidden technical debt in machine learning systems*. In *Advances in Neural Information Processing Systems* (pp. 2503–2511).

[8] Venu Madhav Aragani, Arunkumar Thirunagalingam, "Leveraging Advanced Analytics for Sustainable Success: The Green Data Revolution," in Driving Business Success Through Eco-Friendly Strategies, IGI Global, USA, pp. 229- 248, 2025.

[9] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). *PyTorch: An imperative style, high-performance deep learning library*. In *Advances in Neural Information Processing Systems*, 32.

[10] Lakshmi Narasimha Raju Mudunuri, Pronaya Bhattacharya, "Ethical Considerations Balancing Emotion and Autonomy in AI Systems," in Humanizing Technology With Emotional Intelligence, IGI Global, USA, pp. 443-456, 2025.

[11] Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., & Zhang, G. (2018). *Learning under concept drift: A review*. *IEEE Transactions on Knowledge and Data Engineering*, 31(12), 2346–2363.

[12] S. Panyaram, "Connected Cars, Connected Customers: The Role of AI and ML in Automotive Engagement," International Transactions in Artificial Intelligence, vol. 7, no. 7, pp. 1-15, 2023.

[13] Susmith Barigidad. "Edge-Optimized Facial Emotion Recognition: A High-Performance Hybrid Mobilenetv2-Vit Model". IJAIBDCMS [International JournalofAI,BigData,ComputationalandManagement Studies]. 2025 Apr. 3 [cited 2025 Jun. 4]; 6(2):PP. 1-10.

[14] Zaharia, M., Das, T., Li, H., Hunter, T., Shenker, S., & Stoica, I. (2013). *Discretized streams: Fault-tolerant streaming computation at scale*. In *Proceedings of the 24th ACM Symposium on Operating Systems Principles* (pp. 423–438).

[15] Pulivarthy, P. (2023). ML-driven automation optimizes routine tasks like backup and recovery, capacity planning and database provisioning. Excel International Journal of Technology, Engineering and Management, 10(1), 22–31. https://doi.uk.com/7.000101/EIJTEM

[16] Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). *A survey on concept drift adaptation*. *ACM Computing Surveys (CSUR)*, 46(4), 1–37.

[17] Praveen Kumar Maroju, "Assessing the Impact of AI and Virtual Reality on Strengthening Cybersecurity Resilience Through Data Techniques," Conference: 3rd International conference on Research in Multidisciplinary Studies Volume: 10, 2024

[18] Kuznetsov, S., & Nivargi, R. (2020). *Delta Lake: High-performance ACID table storage over cloud object stores*. *Data Engineering*, 43(1), 45–56.

[19] Mohanarajesh Kommineni (2024) "Investigate Methods for Visualizing the Decision-Making Processes of a Complex AI System, Making Them More Understandable and Trustworthy in financial data analysis" International Transactions in Artificial Intelligence, Pages 1-21

[20] Sato, R., Lin, Y., Shibata, Y., & Ohsuga, A. (2021). *Reproducible machine learning with Pachyderm: Data versioning and pipeline orchestration*. In *IEEE International Conference on Big Data* (pp. 3029–3038).

[21] Tolosana-Calasanz, R., Rana, O. F., & Parashar, M. (2022). *Data provenance and versioning for trustworthy AI*. *Future Generation Computer Systems*.

[22] Bhagath Chandra Chowdari Marella, "From Silos to Synergy: Delivering Unified Data Insights across Disparate Business Units", International Journal of Innovative Research in Computer and Communication Engineering, vol.12, no.11, pp. 11993-12003, 2024.

[23] Divya K, “Efficient CI/CD Strategies: Integrating Git with automated testing and deployment”, World Journal of Advanced Research and Reviews: an International ISSN Approved Journal, vol.20, no.2, pp. 1517-1530, 2023.

[24] A. K. K, G. C. Vegineni, C. Suresh, B. C. Chowdari Marella, S. Addanki and P. Chimwal, "Development of Multi Objective Approach for Validation of PID Controller for Buck Converter," *2025 First International Conference on Advances in Computer Science, Electrical, Electronics, and Communication Technologies (CE2CT)*, Bhimtal, Nainital, India, 2025, pp. 1186-1190, doi: 10.1109/CE2CT64011.2025.10939724.

[25] Pugazhenthi, V. J., Pandy, G., Jeyarajan, B., & Murugan, A. (2025, March). AI-Driven Voice Inputs for Speech Engine Testing in Conversational Systems. In *SoutheastCon 2025* (pp. 700-706). IEEE.

[26] Kiran Nittur, Srinivas Chippagiri, Mikhail Zhidko, “Evolving Web Application Development Frameworks: A Survey of Ruby on Rails, Python, and Cloud-Based Architectures”, International Journal of New Media Studies (IJNMS), 7 (1), 28-34, 2020.

[27] Puvvada, R. K. "Optimizing Financial Data Integrity with SAP BTP: The Future of Cloud-Based Financial Solutions." *European Journal of Computer Science and Information Technology* 13.31 (2025): 101-123.

[28] Designing Of Sepic Pfc Based Plug-In Electric Vehicle Charging Station, Sree Lakshmi Vineetha Bitragunta, International Journal of Core Engineering & Management, Volume-7, Issue-01, 2022, PP-233-242.

[29] Animesh Kumar, “AI-Driven Innovations in Modern Cloud Computing”, Computer Science and Engineering, 14(6), 129-134, 2024.

[30] Kirti Vasdev. (2019). “GIS in Disaster Management: Real-Time Mapping and Risk Assessment”. International Journal on Science and Technology, 10(1), 1–8. https://doi.org/10.5281/zenodo.14288561

[31] Noor, S., Awan, H.H., Hashmi, A.S. et al. “Optimizing performance of parallel computing platforms for large-scale genome data analysis”. Computing 107, 86 (2025). https://doi.org/10.1007/s00607-025-01441-y.

[32] Venkata Nagendra Kumar Kundavaram, Venkata Krishna Reddy Kovvuri, Krishna Prasanth Brahmaji Kanagarla. Data Quality Evaluation Framework For High-Volume Database Systems. International Journal of Engineering Development and Research.(2025)13(3), 209-218.

[33] Venkata SK Settibathini. Optimizing Cash Flow Management with SAP Intelligent Robotic Process Automation (IRPA). Transactions on Latest Trends in Artificial Intelligence, 2023/11, 4(4), PP 1-21, https://www.ijsdcs.com/index.php/TLAI/article/view/469/189