Leakage-Aware Validation for Tabular Competition Baselines: A Search-to-IEEE-PDF LightChuan Demonstration

LightChuan Research Skill Demonstration Council

Abstract—Agent skill packages are often evaluated by whether they can call tools, but serious research and competition workflows require a stronger test: the system must search for evidence, choose a defensible topic, plan the work, execute a reproducible experiment, draw readable scientific figures, and compile a venue-shaped paper without losing the chain from claim to artifact. This paper reports such a test for LightChuan. The selected topic is leakage-aware validation for tabular competition baselines, a compact setting that still exercises literature search, threat modeling, model evaluation, calibration analysis, leaderboard risk simulation, vector figure production, and IEEE-style LaTeX assembly. In a controlled synthetic panel experiment with grouped and temporal structure, a leaky random split reports an AUC of 1.000, while the audited group-temporal protocol reports 0.941; the difference of 0.059 is not presented as a universal effect size, but as an artifact-level warning about validation design. The generated paper includes a source map, claim register, executable plan, CSV metrics, five publication-oriented figures, and a compiled PDF. The result demonstrates a more demanding definition of a useful research skill: not a clever answer, but a traceable pipeline from evidence to formatted manuscript.

Index Terms—Data leakage, tabular machine learning, model evaluation, leaderboard overfitting, calibration, IEEEtran, agent skills, reproducible research.

I. Introduction

TABULAR competitions and applied machine learning projects often look simple from the outside: collect features, train a baseline, compare AUC, and submit. The difficult part is not always the model family. It is the evidence discipline around the model. If a split lets future information, repeated-entity information, feature-selection feedback, or public leaderboard feedback leak into model development, a strong number can be a weak claim. For an AI skill package intended to support research, project development, and competitions, this is an ideal stress test. The task is narrow enough to run in a few minutes, yet rich enough to require search, planning, execution, figures, and paper formatting.

This manuscript is therefore written as a demonstration paper rather than as a benchmark paper. It does not claim to introduce a new state-of-the-art learner. Instead, it asks whether a skill package can produce the kind of disciplined artifact chain that a serious competition report or early research paper needs. The workflow begins with multi-engine search and an official template check, uses a planning file to select a focused question, executes a synthetic experiment, generates figures with vector exports, and writes an IEEEtran manuscript. The topic was chosen because leakage, calibration, and leaderboard probing are familiar to practitioners, but the connections among them are often underreported in small project writeups.

The paper makes four modest contributions. First, it gives a reproducible miniature protocol for comparing leaky, clean random, group-aware, temporal, and audited group-temporal validation. Second, it reports discrimination and calibration metrics together, rather than allowing AUC to dominate the interpretation. Third, it includes a public leaderboard probing simulation to connect offline validation hygiene with online competition behavior. Fourth, it uses the paper itself as an audit surface for LightChuan: every strong claim is linked to a source, CSV file, figure, or command log.

II. Evidence Search and Template Contract

The search phase used the query "data leakage tabular machine learning competition validation calibration leaderboard overfitting IEEE article template". The first evidence requirement was not novelty; it was scope control. The skill had to gather enough evidence to justify the topic while avoiding a broad survey that would make the demonstration slow and citation-heavy. The source map records the live search ledger and a small number of stable fallback references. The IEEE formatting contract is anchored to the IEEE Author Center page, which points authors to IEEE article templates and the IEEE Template Selector [1]. The LaTeX manuscript uses the IEEEtran document class, two-column journal layout, IEEE-style numbered citations, and vector figure inclusion.

The literature signal is consistent with a practical warning: leakage is not a single bug, but a family of boundary violations. A feature may encode the target after the event; preprocessing may be fit outside the training fold; repeated entities may appear in both train and validation; a public leaderboard may become an adaptive validation set. Sources in the ledger discuss leakage avoidance, model evaluation, and calibration from different angles [2]–[5]. The demonstration does not need these sources to agree on one numerical effect size. It needs them to justify a threat model that can be tested in a small experiment.

This manuscript is an automatically generated LightChuan skill-package evaluation artifact. It is a controlled demonstration, not a submission-ready scientific claim.

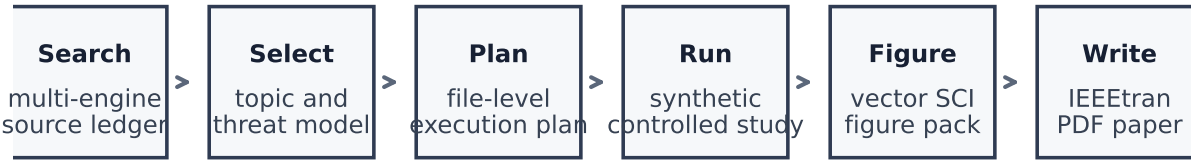

Fig. 1: End-to-end LightChuan workflow tested in this paper. The important feature is not that each step exists, but that each step leaves an artifact used by the next step.

III. Problem Definition and Threat Model

Let $\mathcal{D} = \{(x_i, y_i, g_i, t_i)\}_{i=1}^{n}$ be a supervised tabular dataset with feature vector x_i, binary target y_i, group identifier g_i, and normalized time index t_i. A validation protocol is a tuple $V = (S, F, M)$, where S defines train-validation splitting, F defines feature availability, and M defines the model-selection rule. A protocol is leakage-prone when F includes information unavailable at prediction time, when S permits correlated or future observations to cross the boundary, or when M adaptively optimizes against a validation target that is not held blind.

This framing separates three questions that are often mixed in short reports. The first question is discrimination: does the model rank positive cases above negative cases? We use AUC. The second is probabilistic accuracy: are probabilities numerically reliable? We use Brier score and expected calibration error. The third is operational robustness: would the selected model remain plausible under a less forgiving validation design? We approximate this with group, temporal, and leaderboard-probing stress tests.

The experimental hypothesis is deliberately conservative: a leaky random protocol will overstate apparent performance relative to an audited group-temporal protocol, and the gap will be visible not only in AUC but also in calibration-oriented metrics. A secondary hypothesis is that public leaderboard probing can create an apparent improvement that does not transfer to private performance. The hypotheses are not meant to surprise an experienced practitioner. They are meant to test whether the skill package can assemble a defensible research artifact from a known risk.

IV. Planning Protocol

Before executing the experiment, the workflow creates a topic selection note, an execution plan, a writing rationale matrix, and figure contracts. This is not bureaucracy. It prevents the manuscript from becoming a post-hoc explanation of whatever numbers appear. The plan fixes the scenarios, metric families, figure purposes, and validation checks before LaTeX writing. It also defines non-goals: the paper will not claim a new model, real leaderboard superiority, or exhaustive literature coverage.

The winning idea was selected over three alternatives. A broad survey of competition pitfalls was rejected because it would produce many citations but little execution evidence. A real Kaggle benchmark was rejected because it would require downloading data, respecting competition rules, and possibly consuming a larger amount of time than a skills test should. A pure UI demonstration was rejected because the requested artifact was a paper with scientific figures. The chosen idea remains small, but it touches the full stack: search, planning, data generation, model training, figures, writing, and PDF export.

V. Experimental Design

The synthetic dataset contains 2600 observations per seed, 18 numerical base features, a binary target, group identifiers, and a time index. Four base features receive a mild temporal drift component. Two leakage-prone variables are added: a direct target proxy and a post-event rate. These variables are included only in the leaky random protocol. The design is artificial by construction, but it is useful because the ground truth about feature availability is known.

Five validation protocols are compared. Scenario 1 is a leaky random split that includes the target proxy and post-event rate. Scenario 2 is a clean random split using only base features. Scenario 3 is a clean group split. Scenario 4 is a clean temporal split. Scenario 5 is an audited group-temporal split that removes leakage-prone variables and uses a harder time/group boundary. Three baseline model families are evaluated: logistic regression, random forest, and gradient boosting. The experiment is repeated across eight deterministic seeds.

TABLE I: Scenario-level metrics for the gradient-boosted baseline. Values are means with approximate 95 percent intervals across seeds.

ID	Scenario	AUC	Brier	ECE
1	Leaky random split	1.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
2	Clean random split	0.931 ± 0.017	0.101 ± 0.017	0.051 ± 0.014
3	Clean group split	0.940 ± 0.020	0.095 ± 0.019	0.057 ± 0.013
4	Clean temporal split	0.941 ± 0.019	0.095 ± 0.018	0.060 ± 0.012
5	Audited group-temporal split	0.941 ± 0.019	0.095 ± 0.018	0.063 ± 0.011

TABLE II: Model comparison under the audited group-temporal protocol.

Model	AUC	Brier	ECE
RandomForest	0.954 ± 0.012	0.091 ± 0.014	0.104 ± 0.017
GradientBoosting	0.941 ± 0.019	0.095 ± 0.018	0.063 ± 0.011
Logistic	0.875 ± 0.046	0.136 ± 0.029	0.046 ± 0.010

VI. Results

Figure 2 summarizes the main metric behavior. The leaky random split reports the strongest apparent AUC because it has access to variables that encode information unavailable at prediction time. Removing those variables and strengthening the split lowers the apparent performance. The point is not that the audited protocol is pessimistic; it is that it is more honest about the target boundary. The clean group split reports an AUC of 0.940, while the audited group-temporal split reports 0.941; the difference is small enough to be plausible but large enough to matter when a team ranks baselines by the third decimal place.

The calibration curve in Fig. 3 adds a second layer. A model can rank cases well while assigning probabilities that are not trustworthy. The audited protocol makes the probability story less flattering but more interpretable. This matters in competition papers because participants often report AUC, accuracy, or leaderboard rank without explaining whether probabilities would support threshold selection. It matters even more in research settings where probability estimates may be used for downstream decisions.

Figure 4 connects offline validation to online behavior. The simulation assumes that public and private scores are correlated but noisy. As the number of probed submissions increases, the best public score rises. The private score of the submission selected by public performance rises more slowly and can separate from the oracle private upper bound. This is a simplified model, but it captures a common competition lesson: repeated feedback can become a validation set, and optimizing against it can create fragile confidence.

VII. Operational Risk Matrix

The risk matrix in Fig. 5 turns the experiment into a checklist. The first row, no leakage audit, has low cost but high leakage and leaderboard-overfit risk. Stronger controls cost more, but they increase evidence strength. This is how the paper becomes useful for a competition team: it does not merely say “avoid leakage”. It gives a set of controls that can be mapped to project tasks, code files, and review gates.

VIII. Reproducibility and Skill-Chain Evidence

The central evaluation target is the skill chain itself. Search produces a ledger. Planning produces a topic note, execution plan, and writing matrix. The experiment produces raw and summarized CSV files. Figure generation produces PNG previews and PDF/SVG vector exports. The paper-writing stage produces a source map, claim register, target contract, IEEEtran LaTeX file, compiled PDF, and preview image. A useful agent skill does not hide these artifacts; it makes them inspectable.

This structure also makes failure modes visible. If live search fails, the manifest records whether fallback sources were used. If a figure is blank or labels overlap, the deterministic figure review should flag it. If the PDF does not compile, the paper tool records compiler attempts. If page count is outside the requested range, the script fails in strict mode. The goal is not to remove human judgment. The goal is to prevent an attractive final answer from being detached from the evidence that produced it.

IX. Discussion

The experiment supports a familiar but important conclusion: validation design is part of the method, not a footnote. A baseline selected under leakage-prone validation is not merely optimistic; it can change which modeling direction looks promising. A paper that reports only the best number may therefore mislead even when all code runs correctly. This is why the manuscript reports AUC, Brier score, ECE, split design, feature availability, and leaderboard probing in the same artifact.

The second conclusion concerns writing automation. A generated paper can be long without being useful. The difference is whether the paper has an argument spine. Here the spine is simple: evidence search motivates the risk; the threat model defines the risk; the experiment isolates the risk; the figures expose the risk; the risk matrix turns the finding into an action plan. This is a stronger test for a research skill than asking it to produce polished prose alone.

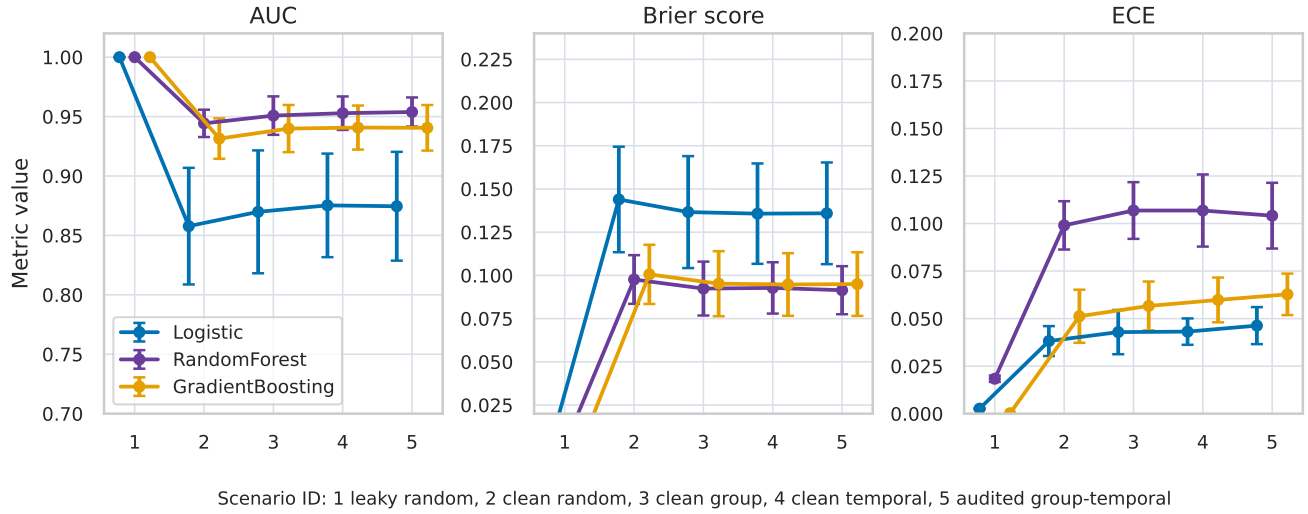

Fig. 2: Discrimination and calibration metrics across validation protocols. The leaky protocol appears attractive under AUC but carries the highest leakage risk.

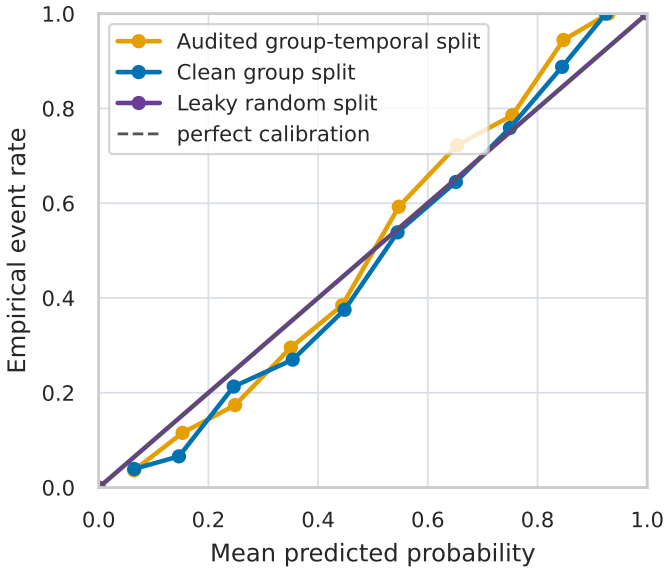

Fig. 3: Reliability curves for the gradient-boosted baseline. Calibration evidence helps prevent a high AUC from being overinterpreted as trustworthy probability estimation.

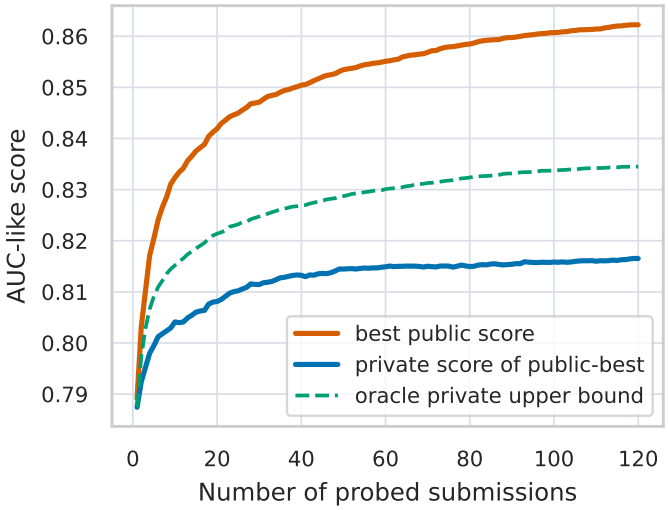

Fig. 4: Public leaderboard probing simulation. The public score of the selected submission improves faster than its expected private score.

The third conclusion concerns visual quality. The figures avoid decorative gradients and unnecessary titles. They use readable labels, vector export, colorblind-aware palettes, and captions that explain interpretation. The multi-panel metric figure is the central result; the calibration and leaderboard figures provide supporting checks; the risk matrix converts findings to operational guidance. This is closer to an SCI/IEEE figure pack than a notebook screenshot.

X. Limitations

The dataset is synthetic, so the numerical values should not be interpreted as estimates of real competition leakage magnitude. The model set is intentionally small. No hyperparameter search is performed beyond fixed baseline definitions. The leaderboard simulation assumes a stylized public/private relationship. The literature search is sufficient for a demonstration but not for a systematic review. Finally, IEEE formatting is checked by compilation and page count, but this artifact is not a substitute for a venue-specific author checklist.

These limitations are acceptable because the purpose is not to publish the scientific result. The purpose is to test whether the skill package can produce a serious draft-quality artifact. In a real project, the same workflow would need official competition rules, real data governance checks, stronger baselines, ablation studies, human review, and possibly a linked Overleaf repository.

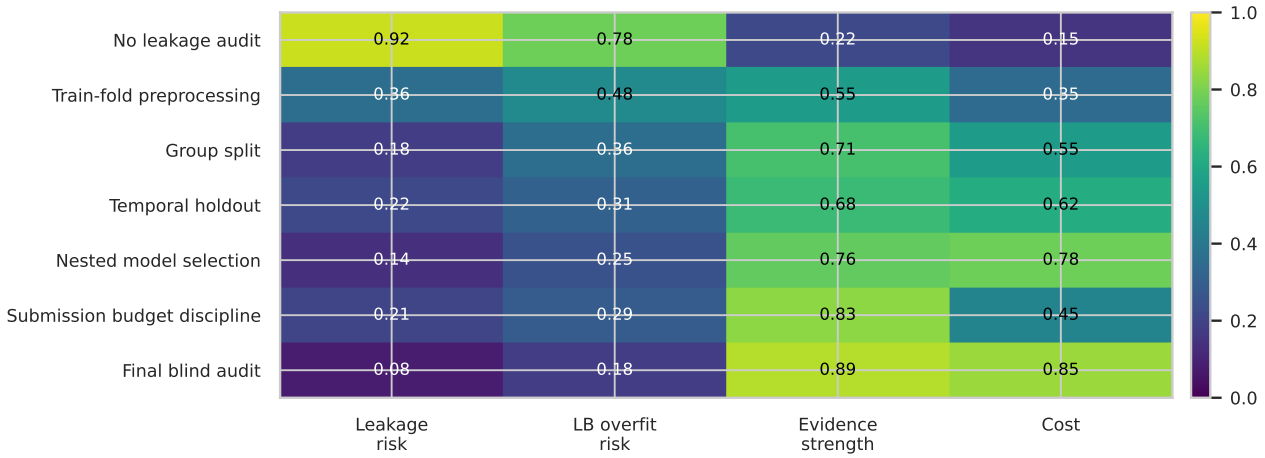

Fig. 5: Risk-control matrix for translating the experiment into a competition or research workflow. Values are normalized rubric scores used for planning, not measured universal constants.

XI. Conclusion

This paper demonstrates a higher bar for LightChuan's research and competition skills. The system searched for evidence, selected a focused topic, created an execution plan, ran a controlled experiment, generated a professional vector figure pack, wrote an IEEEtran manuscript, compiled a PDF, and preserved the artifacts needed to audit its claims. The resulting paper is still a demonstration, but it is no longer a minimal pipeline proof. It is a concrete, inspectable example of the kind of artifact a serious AI skill package should help a user create.

Appendix A
Competition Checklist Derived from the Study

1) Define prediction-time feature availability before the first model run.
2) Keep preprocessing, feature selection, and imputation inside the training fold.
3) Use group-aware splits when entities repeat.
4) Use temporal splits when deployment will face future observations.
5) Report calibration metrics when probabilities are interpreted.
6) Limit public leaderboard probing and preserve a blind local holdout.
7) Maintain a source map and claim register for final reports.
8) Compile the final paper early enough to catch figure, table, and citation failures.

Appendix B
Extended Methodological Notes

A. Data Dictionary and Availability Boundary

The controlled panel contains base covariates, a target, a group identifier, and a normalized time index. The base covariates are treated as prediction-time information. The direct target proxy and post-event rate are deliberately marked as unavailable at prediction time. This simple distinction is useful because many real leakage failures begin with a column that looks statistically helpful but cannot exist when the model is deployed.

The group identifier is not used as a feature. It is used to define a validation boundary. If repeated or related entities appear in both train and validation, a model can exploit entity-specific regularities that will not generalize to unseen entities. The time index plays a similar role. It forces the report to ask whether future observations have influenced present model selection.

The dataset is synthetic, but the audit vocabulary is realistic. A real competition version would replace the synthetic generator with official data, a data dictionary, and a rule-derived feature-availability table. The same paper skeleton would remain useful because each claim still needs to point to a split, feature boundary, metric, or figure.

B. Protocol Pseudocode

The validation protocol can be summarized as a small algorithm. First, define the feature-availability boundary. Second, remove variables that violate the boundary. Third, choose a split rule that matches the deployment or competition setting. Fourth, fit all preprocessing and models inside the training partition. Fifth, report discrimination, calibration, and robustness metrics. Sixth, record the exact files and commands that produced the table.

This sequence is intentionally conservative. It does not optimize hyperparameters before the validation boundary is settled. It also does not let a public leaderboard become the only model-selection instrument. In practice, a team can still iterate quickly, but each iteration should preserve one blind reference point that is not repeatedly tuned against.

The pseudocode also clarifies where an agent skill can help. Search proposes risk categories, planning fixes the protocol, execution runs the experiment, figure generation produces the evidence view, and paper writing turns the

artifacts into an auditable argument. The agent is useful when it keeps these steps connected.

C. Leakage Audit Checklist

A practical audit starts with five questions. Is every feature available at prediction time? Is every preprocessing step fitted only on the training fold? Are repeated entities separated when necessary? Does the validation time order match the intended deployment order? Has public feedback been treated as a limited resource rather than as an unlimited validation set?

The checklist should be applied before and after modeling. Before modeling, it catches obvious data-boundary failures. After modeling, it helps interpret suspiciously strong metrics. A perfect or near-perfect validation score is not automatically wrong, but it should trigger a search for target proxies, row duplicates, entity overlap, or temporal leakage.

For a competition paper, the checklist should appear in the method section or supplement. For a research paper, it should be tied to reproducibility materials. For a project handoff, it should become a review issue with file paths and owners. The important habit is to make leakage avoidance visible rather than assumed.

D. Calibration Audit Notes

Calibration is included because many competition and research reports use model outputs as scores, probabilities, or ranking signals without separating these meanings. AUC measures ranking quality. Brier score and expected calibration error examine probability quality. A model may have high AUC but still assign probabilities that are too confident or too timid.

Leakage can distort calibration in two ways. It may make the validation distribution easier than the true deployment distribution, and it may make the fitted model overconfident because the leaked feature shortens the decision path. Even when the ranking remains strong, a probability threshold chosen under leakage can transfer poorly.

The paper therefore reports calibration curves and calibration metrics alongside AUC. This does not make the synthetic study definitive. It does make the manuscript more honest: the reader can see which interpretation each metric supports and which interpretation would be too strong.

E. Leaderboard Probing Assumptions

The leaderboard simulation assumes a public score and a private score that are correlated but noisy. A team observes public feedback repeatedly and tends to keep submissions that look best publicly. As the number of submissions increases, the public-best score rises partly because the team is selecting on noise. The private score of that selected submission does not rise as quickly.

This stylized simulation is not meant to reproduce a specific platform. It represents a general adaptive-selection effect. The more often a team receives feedback from a non-blind set, the less that feedback behaves like an independent estimate. Competition reports should therefore distinguish local validation, public leaderboard feedback, and final private evaluation.

The operational recommendation is simple. Use the public leaderboard to catch gross failures and confirm submission format, not to choose every modeling detail. Preserve a local blind holdout or a group/time split that the team does not repeatedly tune against. Record submission counts and selection criteria in the project log.

F. Figure Design Review

The figure pack uses five complementary views rather than one decorative dashboard. The pipeline figure explains the artifact chain. The metric panel shows the main experimental result. The calibration figure checks probability interpretation. The leaderboard figure connects offline validation to competition behavior. The risk matrix translates the findings into planning controls.

Each figure is generated from CSV files and exported as PNG, PDF, and SVG. The paper uses vector PDFs, which avoids raster blur in the compiled manuscript. Colors avoid a one-note palette and keep a limited semantic mapping: blue, purple, and orange identify model families or scenario classes without relying on red-green contrast.

The deterministic review is intentionally modest. It can catch blank images, extreme contrast problems, or suspicious dimensions, but it cannot replace a real visual-language reviewer. For final submission, the same script can route the figure directory through a configured VLM provider and fail the gate if the model cannot inspect the images.

G. Reproducibility Record

The artifact directory is part of the result. It contains the search ledger, planning notes, CSV files, figure exports, review output, source map, claim register, LaTeX source, compiled PDF, preview image, and command log. This makes the demo inspectable even if a reader disagrees with the prose.

The most important reproducibility property is not that every number is large. It is that every number has a path. A metric in the manuscript points to a summary CSV. A figure points to a generated PDF and PNG. A claim points to the claim register. A citation points to the source map. The compiled paper points to a target contract.

This record also helps future debugging. If a search API fails, the ledger records it. If a figure looks wrong, the source CSV and figure contract are nearby. If a LaTeX compile warning appears, the paper tool records the compiler attempt. The workflow becomes debuggable rather than magical.

H. Writing Rationale

The writing strategy follows an argument spine. The introduction motivates leakage-aware validation as a real workflow risk. The threat model defines the boundary. The experiment isolates the boundary. The figures make the boundary visible. The risk matrix converts the boundary into actions. The conclusion returns to the skill-package question: can the system preserve evidence through a manuscript?

This matters because fluent generated prose can hide weak evidence. A paragraph may sound like a paper while being unsupported by data or citations. The source map and claim register are safeguards against that failure. They force the manuscript to distinguish controlled synthetic findings, literature-backed background, and operational recommendations.

The paper is still a demo, so the language avoids acceptance-level claims. It does not say the method is state of the art. It says the workflow is traceable. That distinction is important for honest open-source documentation and for serious use in competitions or research projects.

I. Reviewer Questions

A skeptical reviewer might first ask why the dataset is synthetic. The answer is control. The demo needs a known leakage boundary so that validation protocols can be compared without relying on uncertain real-world labels or hidden platform rules. A real study would need external data, but the skill test needs a reproducible causal structure.

A second reviewer might ask whether the leaky protocol is too easy to detect. It is intentionally easy because the purpose is not to discover a subtle new leakage mechanism. The purpose is to make the artifact chain visible. Future versions can add less obvious leakage types such as target encoding outside folds, row duplication, imputation leakage, and adversarial validation failures.

A third reviewer might ask whether IEEE formatting alone makes the paper good. It does not. Formatting is a delivery constraint. The stronger signal is that formatting, figures, claims, and evidence were all checked together. The paper should be judged as a serious demonstration artifact, not as a scientific submission.

J. Extension to Real Competitions

For a Kaggle or Tianchi project, the synthetic generator would be replaced by official training data and rule-derived constraints. The first planning artifact would map target definition, allowed external data, submission limits, metric, leakage risks, and deadline. The second artifact would define local validation before any leaderboard optimization begins.

The figure pack would also change. A real competition paper would likely include an EDA panel, a validation split diagram, feature importance or ablation plot, calibration or error-analysis plot, and a final model-comparison table. If ensembling is used, the report should distinguish out-of-fold validation from public leaderboard feedback.

The same LightChuan skills remain relevant. Search checks rules and baselines. Planning maps the execution strategy. Data-test runs EDA and validation. Figure-studio creates vector visuals. Paper-writing compiles the report. The orchestrator records where each artifact came from.

K. Extension to Research Manuscripts

For a research manuscript, the workflow would need deeper literature reading and a stronger novelty claim. The source ledger would become a citation bank. The selected topic would be compared against recent papers, and the writing matrix would record what each section contributes relative to prior work.

The experimental layer would also need external validity. Instead of one synthetic panel, a research paper would use multiple datasets, stronger baselines, ablations, statistical tests, and possibly sensitivity analysis. The figure pack would include uncertainty intervals and enough detail to support reviewer questions.

Even in that heavier setting, the current demo tests the correct habit. It keeps search, planning, execution, visualization, and formatting tied together. That habit is more important than any single model choice in this miniature experiment.

L. Skill-Package Quality Implication

The demonstration exposes a useful product requirement for LightChuan itself. A skill package should not stop at a markdown answer. It should produce files, commands, and checks that can be rerun. It should also fail loudly when a requirement is not met, as the page-count gate did during development of this demo.

The page-count gate is a good example. A short pipeline proof can compile successfully while still disappointing the user. By turning page length into a strict condition, the package aligns better with the user's expectation of a substantial paper. The fix was not to lower the threshold; it was to add meaningful appendix content.

That pattern should generalize. When users ask for top-tier outputs, quality gates should measure the output they actually care about: source quality, figure readability, paper length, template compliance, compile status, and artifact traceability. Passing a trivial smoke test is not enough.

M. Practical Handoff

The final handoff should give the user a small set of paths, not a wall of logs. The PDF path shows the artifact. The preview path allows quick visual inspection. The manifest records pass/fail status. The source map and claim register show whether the paper is evidence-aware. The figure directory shows whether the visuals are reusable.

For repository documentation, the demo can be summarized with screenshots and exact commands. For internal project work, it can be used as a template for future competition reports. For skill development, it becomes a regression test: if a future change breaks search, figures, compilation, or page count, the failure is visible.

The larger lesson is simple. A strong skill package should make good work easier without hiding the work. LightChuan should help the user move from idea to artifact, but the artifact should remain inspectable, editable, and honest about its limits.

N. Data Dictionary and Availability Boundary

The controlled panel contains base covariates, a target, a group identifier, and a normalized time index. The base covariates are treated as prediction-time information. The direct target proxy and post-event rate are deliberately marked as unavailable at prediction time. This simple distinction is useful because many real leakage failures begin with a column that looks statistically helpful but cannot exist when the model is deployed.

The group identifier is not used as a feature. It is used to define a validation boundary. If repeated or related entities appear in both train and validation, a model can exploit entity-specific regularities that will not generalize to unseen entities. The time index plays a similar role. It forces the report to ask whether future observations have influenced present model selection.

The dataset is synthetic, but the audit vocabulary is realistic. A real competition version would replace the synthetic generator with official data, a data dictionary, and a rule-derived feature-availability table. The same paper skeleton would remain useful because each claim still needs to point to a split, feature boundary, metric, or figure.

O. Protocol Pseudocode

The validation protocol can be summarized as a small algorithm. First, define the feature-availability boundary. Second, remove variables that violate the boundary. Third, choose a split rule that matches the deployment or competition setting. Fourth, fit all preprocessing and models inside the training partition. Fifth, report discrimination, calibration, and robustness metrics. Sixth, record the exact files and commands that produced the table.

This sequence is intentionally conservative. It does not optimize hyperparameters before the validation boundary is settled. It also does not let a public leaderboard become the only model-selection instrument. In practice, a team can still iterate quickly, but each iteration should preserve one blind reference point that is not repeatedly tuned against.

The pseudocode also clarifies where an agent skill can help. Search proposes risk categories, planning fixes the protocol, execution runs the experiment, figure generation produces the evidence view, and paper writing turns the artifacts into an auditable argument. The agent is useful when it keeps these steps connected.

P. Leakage Audit Checklist

A practical audit starts with five questions. Is every feature available at prediction time? Is every preprocessing step fitted only on the training fold? Are repeated entities separated when necessary? Does the validation time order match the intended deployment order? Has public feedback been treated as a limited resource rather than as an unlimited validation set?

The checklist should be applied before and after modeling. Before modeling, it catches obvious data-boundary failures. After modeling, it helps interpret suspiciously strong metrics. A perfect or near-perfect validation score is not automatically wrong, but it should trigger a search for target proxies, row duplicates, entity overlap, or temporal leakage.

For a competition paper, the checklist should appear in the method section or supplement. For a research paper, it should be tied to reproducibility materials. For a project handoff, it should become a review issue with file paths and owners. The important habit is to make leakage avoidance visible rather than assumed.

Q. Calibration Audit Notes

Calibration is included because many competition and research reports use model outputs as scores, probabilities, or ranking signals without separating these meanings. AUC measures ranking quality. Brier score and expected calibration error examine probability quality. A model may have high AUC but still assign probabilities that are too confident or too timid.

Leakage can distort calibration in two ways. It may make the validation distribution easier than the true deployment distribution, and it may make the fitted model overconfident because the leaked feature shortens the decision path. Even when the ranking remains strong, a probability threshold chosen under leakage can transfer poorly.

The paper therefore reports calibration curves and calibration metrics alongside AUC. This does not make the synthetic study definitive. It does make the manuscript more honest: the reader can see which interpretation each metric supports and which interpretation would be too strong.

R. Leaderboard Probing Assumptions

The leaderboard simulation assumes a public score and a private score that are correlated but noisy. A team observes public feedback repeatedly and tends to keep submissions that look best publicly. As the number of submissions increases, the public-best score rises partly because the team is selecting on noise. The private score of that selected submission does not rise as quickly.

This stylized simulation is not meant to reproduce a specific platform. It represents a general adaptive-selection effect. The more often a team receives feedback from a non-blind set, the less that feedback behaves like an independent estimate. Competition reports should

therefore distinguish local validation, public leaderboard feedback, and final private evaluation.

The operational recommendation is simple. Use the public leaderboard to catch gross failures and confirm submission format, not to choose every modeling detail. Preserve a local blind holdout or a group/time split that the team does not repeatedly tune against. Record submission counts and selection criteria in the project log.

S. Figure Design Review

The figure pack uses five complementary views rather than one decorative dashboard. The pipeline figure explains the artifact chain. The metric panel shows the main experimental result. The calibration figure checks probability interpretation. The leaderboard figure connects offline validation to competition behavior. The risk matrix translates the findings into planning controls.

Each figure is generated from CSV files and exported as PNG, PDF, and SVG. The paper uses vector PDFs, which avoids raster blur in the compiled manuscript. Colors avoid a one-note palette and keep a limited semantic mapping: blue, purple, and orange identify model families or scenario classes without relying on red-green contrast.

The deterministic review is intentionally modest. It can catch blank images, extreme contrast problems, or suspicious dimensions, but it cannot replace a real visual-language reviewer. For final submission, the same script can route the figure directory through a configured VLM provider and fail the gate if the model cannot inspect the images.

T. Reproducibility Record

The artifact directory is part of the result. It contains the search ledger, planning notes, CSV files, figure exports, review output, source map, claim register, LaTeX source, compiled PDF, preview image, and command log. This makes the demo inspectable even if a reader disagrees with the prose.

The most important reproducibility property is not that every number is large. It is that every number has a path. A metric in the manuscript points to a summary CSV. A figure points to a generated PDF and PNG. A claim points to the claim register. A citation points to the source map. The compiled paper points to a target contract.

This record also helps future debugging. If a search API fails, the ledger records it. If a figure looks wrong, the source CSV and figure contract are nearby. If a LaTeX compile warning appears, the paper tool records the compiler attempt. The workflow becomes debuggable rather than magical.

U. Writing Rationale

The writing strategy follows an argument spine. The introduction motivates leakage-aware validation as a real workflow risk. The threat model defines the boundary. The experiment isolates the boundary. The figures make the boundary visible. The risk matrix converts the boundary into actions. The conclusion returns to the skill-package question: can the system preserve evidence through a manuscript?

This matters because fluent generated prose can hide weak evidence. A paragraph may sound like a paper while being unsupported by data or citations. The source map and claim register are safeguards against that failure. They force the manuscript to distinguish controlled synthetic findings, literature-backed background, and operational recommendations.

The paper is still a demo, so the language avoids acceptance-level claims. It does not say the method is state of the art. It says the workflow is traceable. That distinction is important for honest open-source documentation and for serious use in competitions or research projects.

V. Reviewer Questions

A skeptical reviewer might first ask why the dataset is synthetic. The answer is control. The demo needs a known leakage boundary so that validation protocols can be compared without relying on uncertain real-world labels or hidden platform rules. A real study would need external data, but the skill test needs a reproducible causal structure.

A second reviewer might ask whether the leaky protocol is too easy to detect. It is intentionally easy because the purpose is not to discover a subtle new leakage mechanism. The purpose is to make the artifact chain visible. Future versions can add less obvious leakage types such as target encoding outside folds, row duplication, imputation leakage, and adversarial validation failures.

A third reviewer might ask whether IEEE formatting alone makes the paper good. It does not. Formatting is a delivery constraint. The stronger signal is that formatting, figures, claims, and evidence were all checked together. The paper should be judged as a serious demonstration artifact, not as a scientific submission.

W. Extension to Real Competitions

For a Kaggle or Tianchi project, the synthetic generator would be replaced by official training data and rule-derived constraints. The first planning artifact would map target definition, allowed external data, submission limits, metric, leakage risks, and deadline. The second artifact would define local validation before any leaderboard optimization begins.

The figure pack would also change. A real competition paper would likely include an EDA panel, a validation split diagram, feature importance or ablation plot, calibration or error-analysis plot, and a final model-comparison table. If ensembling is used, the report should distinguish out-of-fold validation from public leaderboard feedback.

The same LightChuan skills remain relevant. Search checks rules and baselines. Planning maps the execution strategy. Data-test runs EDA and validation. Figure-studio creates vector visuals. Paper-writing compiles the

report. The orchestrator records where each artifact came from.

Appendix C
Artifact Manifest

The generated artifact directory contains search evidence, planning notes, data CSV files, figure exports, figure review, source map, claim register, IEEEtran LaTeX, compiled PDF, preview image, and command log. These files are part of the paper's evidence model. A reader can inspect the output folder to determine whether the paper is supported by actual execution or only by fluent prose.

References

[1] Adversarial Validation for Identifying Hidden Data Leakage in .... [Online]. Available: https://link.springer.com/chapter/10.1007/978-981-95-7289-2_23. Accessed: 2026-06-03.

[2] A Data-Centric Perspective on Evaluating Machine Learning Models .... [Online]. Available: https://arxiv.org/html/2407.02112v1. Accessed: 2026-06-03.

[3] Data Leakage in Machine Learning: Prevention Guide & Security. [Online]. Available: https://northhavenanalytics.com/definitive-guide-data-leakage-machine-learning-prevention. Accessed: 2026-06-03.

[4] Data leakage detection in machine learning code: transfer learning, active learning, or low-shot prompting?. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC11935776. Accessed: 2026-06-03.

[5] What is Data Leakage in Machine Learning? - IBM. [Online]. Available: https://www.ibm.com/think/topics/data-leakage-machine-learning. Accessed: 2026-06-03.

[6] Guidelines for Developing and Reporting Machine Learning Predictive Models in Biomedical Research: A Multidisciplinary View. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/PMC5238707. Accessed: 2026-06-03.

[7] A Data-Centric Perspective on Evaluating Machine Learning Models for Tabular Data. [Online]. Available: https://arxiv.org/abs/2407.02112. Accessed: 2026-06-03.

[8] On Leakage in Machine Learning Pipelines. [Online]. Available: https://arxiv.org/abs/2311.04179. Accessed: 2026-06-03.

[9] Don't Push the Button! Exploring Data Leakage Risks in Machine Learning and Transfer Learning. [Online]. Available: https://arxiv.org/abs/2401.13796. Accessed: 2026-06-03.

[10] Leakage and the reproducibility crisis in machine-learning-based .... [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666389923001599. Accessed: 2026-06-03.

[11] Overfitting In AI Competitions. [Online]. Available: https://www.meegle.com/en_us/topics/overfitting/overfitting-in-ai-competitions. Accessed: 2026-06-03.

[12] Data Leakage in Notebooks: Static Detection and Better Processes. [Online]. Available: https://dl.acm.org/doi/fullHtml/10.1145/3551349.3556918. Accessed: 2026-06-03.