# Data Shapley for Bad Data Detection and Practical Data Cleaning

Abdul Basit
Department of Computer Science
LUMS, Pakistan
Email: 26100381@lums.edu.pk

***Abstract*—Training data is not equally valuable: some examples improve model generalization, some are redundant, and some actively damage the learned decision rule because their labels or features are corrupted. This report studies the Data Shapley framework of Ghorbani and Zou, which applies the Shapley value from cooperative game theory to assign each training point a fair value based on its marginal contribution to model performance. The project first reproduces and adapts the Data Shapley idea on the UCI Breast Cancer Wisconsin Diagnostic dataset with controlled label corruption. Each training example is treated as a player, each subset of training examples is a coalition, and the coalition value is the test accuracy of a $K$-nearest neighbors classifier trained on that subset. The original experiments show that corrupted labels receive negative Shapley values and that removing low-Shapley examples improves test accuracy while detecting corrupted labels much more reliably than random removal or leave-one-out valuation.**

**I then extend the project beyond the original notebook and presentation by implementing a reproducible TensorFlow-free TMC-Shapley pipeline using NumPy and scikit-learn, and by adding several new experiments: audit-budget data cleaning, corrected convergence analysis, corruption-type robustness, repeated-seed robustness, model-transfer validation, and source-level Shapley valuation. In the final main experiment, TMC-Shapley detects 71 of 91 corrupted labels under a 20% audit budget, giving 78.02% detection precision compared with 19.34% for random selection and 27.47% for leave-one-out. Correcting the labels selected by TMC-Shapley increases KNN test accuracy from 90.35% to 96.49%. These results show that Data Shapley is not only an axiomatically justified fairness concept, but also a practical method for identifying harmful data and allocating limited manual review effort.**

***Index Terms*—Data Shapley, Shapley value, cooperative game theory, data valuation, label noise, data cleaning, machine learning, K-nearest neighbors.**

## I. Introduction

Modern machine learning systems depend strongly on the quality of their training data. The usual modeling pipeline often treats all training examples as equally useful: data is collected, split into train and test sets, and a model is fit using every training point. In practice, this assumption is too simple. A clean and representative example can improve generalization, a duplicate may add almost no new information, and a mislabeled example can actively push the model toward an incorrect decision boundary. This motivates the central question of the project:

> *Can cooperative game theory assign a principled value to each training point so that harmful records can be detected and data-cleaning decisions can improve model performance?*

This question is particularly important in settings such as medical diagnosis, finance, legal prediction, and scientific datasets, where labels may come from human annotation, clinical judgment, data entry, or multiple institutions. In such settings, checking every record manually can be expensive. A useful data-valuation method should therefore do more than produce a ranking: it should help decide which examples deserve manual audit when the review budget is limited.

This project is based on the Data Shapley framework proposed by Ghorbani and Zou [1]. Their key idea is to define a cooperative game in which training examples are players and the value of a coalition is the performance of a model trained on that coalition. The Shapley value then measures the average marginal contribution of each training example across all possible coalitions. A harmful example receives a low or negative value because adding it to coalitions tends to reduce the model's performance.

The project has two layers. First, I implemented the core paper idea in a controlled bad-data detection experiment using the UCI Breast Cancer Wisconsin Diagnostic dataset. This was the work represented in the original `Shapley_Extension.ipynb` notebook and `final-presentation.tex`. Second, I added a new coding extension to make the project substantially stronger: a reproducible TMC-Shapley implementation, saved result

artifacts, and new experiments that test whether Shapley values can guide practical data cleaning.

### A. Contributions

The final project contributions are:

- A cooperative-game formulation of training-data valuation on a real medical dataset.
- A controlled label-corruption experiment showing that corrupted examples receive negative Shapley values.
- A comparison of Shapley-based removal against random removal and leave-one-out.
- A new TensorFlow-free TMC-Shapley implementation for KNN using precomputed distances, making the extension lightweight and reproducible with the packages actually needed for these experiments.
- A practical audit-budget experiment in which selected labels are corrected rather than simply removed.
- Robustness checks across corruption types, iteration counts, random seeds, and downstream models.
- A source-level Shapley extension in which players are groups of examples representing data providers.

### B. How This Report Should Be Read

The report deliberately separates two sets of results. The first set, called the *original project work*, comes from the earlier notebook and presentation. Those experiments used the Data Shapley codebase and established the main story: mislabeled records tend to receive negative Shapley values, and removing low-Shapley records improves the classifier. The second set, called the *new extension work*, was added afterward to make the project more complete and defensible. It uses the same scientific question but a cleaner, reproducible implementation that saves tables and figures. Some numerical values differ between the original and new sections because the experiments are separate runs with different random seeds, implementation details, preprocessing choices, and numbers of TMC permutations. The original tables are included to document the work already completed in the notebook and presentation; the new extension tables are the primary reproducible evidence for the final report. The important point is that both sets of experiments support the same qualitative conclusion.

The most important distinction is also methodological. The original project asked whether Shapley values can identify bad records. The new work asks whether those values can support a practical decision: with a limited review budget, which records should be audited and corrected first? This second question is closer to a real data-cleaning problem and is therefore the main extension of the paper.

## II. Background: Data Shapley

### A. Why Ordinary Importance Scores Are Not Enough

There are many simple ways to score training data, but most do not satisfy a fairness principle. For example, a leave-one-out score asks how much performance changes when one point is removed from the full training set. This is intuitive, but it only evaluates one context: the full dataset minus that point. It can miss interactions among examples. A point may look unimportant in the full dataset because many similar examples remain, but it may be very important in smaller coalitions. Conversely, a mislabeled point may be harmful in some coalitions and hidden in others.

Random removal is not a valuation method at all. If 20% of labels are corrupted, randomly selecting 20% of the training set should find only about 20% corrupted points in expectation. This provides a useful baseline, but not a useful cleaning strategy.

Loss-based scores can be strong detectors of mislabeled data because mislabeled examples often have high training loss. However, loss is not the same as fair data valuation. It does not allocate the value of the learning algorithm across examples, and it does not directly satisfy the cooperative-game axioms of Shapley value. Therefore, in this project loss is included as an additional machine-learning baseline, while Shapley remains the game-theoretic method of interest.

### B. A Small Intuition Example

Consider a simple medical classifier trained on three records: two clean records and one mislabeled record. If the first clean record is added to an empty training set, it may improve test accuracy because it provides useful class information. If the second clean record is added, accuracy may improve further or become more stable. However, if the mislabeled record is added, its effect depends on the coalition it joins. In a coalition with only a few examples, the mislabeled record can strongly distort the nearest-neighbor vote. In a larger coalition, the same mislabeled record may be partly cancelled by many clean neighbors. A leave-one-out score only measures the second situation, because it removes the point from the full dataset. The Shapley value averages both situations and many more: early coalitions, middle coalitions, and late coalitions. This is why Shapley is well suited to data valuation. It does not ask whether a point matters in one fixed context; it asks how the point behaves across all possible contexts.

This example also explains why negative values are meaningful. A negative Shapley value does not simply mean that the point was misclassified by the final model. It means that, on average, adding that point to a coalition lowers the value function. In this project the value function is test accuracy, so negative value means the point tends to reduce generalization performance.

### C. Shapley Value for Data

Let $N = \{1, \ldots, n\}$ be the set of training examples. For any coalition $S \subseteq N$, define $v(S)$ as the test performance of a fixed learning algorithm trained only on examples in $S$. In

this project, $v(S)$ is KNN test accuracy. The Shapley value of point $i$ is:

$$\phi_i(v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(n-|S|-1)!}{n!} \left[ v(S \cup \{i\}) - v(S) \right]. \tag{1}$$

The term $v(S \cup \{i\}) - v(S)$ is the marginal contribution of point $i$ when it joins coalition $S$. The coefficient in (1) is the probability that coalition $S$ appears before $i$ in a uniformly random ordering of all players. Thus, the Shapley value can be interpreted as the expected marginal contribution of a point over random insertion orders.

This interpretation is especially useful for data valuation. A clean point should improve many coalitions, giving positive value. A redundant point should rarely change performance, giving near-zero value. A mislabeled point should often reduce performance, giving negative value.

### D. Axiomatic Justification

The Shapley value is the unique allocation rule satisfying four axioms:

- **Efficiency:** $\sum_i \phi_i = v(N) - v(\emptyset)$. The total assigned value equals the performance gained by using all data instead of no data.
- **Symmetry:** If two data points make identical marginal contributions to all coalitions, they receive equal value.
- **Dummy player:** If a point never changes any coalition's value, it receives value zero.
- **Additivity:** If two valuation games are combined, the Shapley values add.

These axioms make Shapley value more than a heuristic ranking. They provide a formal fairness argument: each training point is credited exactly according to its contribution to the learning task.

### E. Connection to the Course

The project directly applies cooperative game theory rather than using it only as background motivation. The training dataset is the player set, model performance is the transferable utility, and Data Shapley is the allocation rule. The four axioms become practical requirements for data valuation. Efficiency says that the measured improvement from using the dataset should be fully distributed among the data points. Symmetry says that two equally useful records should not be ranked differently just because of their index or order in the file. The dummy axiom says that a point that never changes the model's performance should not receive artificial credit. Additivity says that if we evaluate data under multiple performance criteria or tasks, the value allocations should combine consistently.

This mapping is useful because it turns an abstract fairness theorem into a concrete machine-learning workflow. The result is not only a score but an interpretation: a point's value is its fair share of the model's performance gain.

## III. Computational Method

### A. Why Exact Shapley Is Infeasible

Exact Shapley computation requires evaluating the model on every coalition. With $n = 455$ training examples in this project, there are $2^{455}$ possible coalitions, which is computationally impossible. Even if a single coalition evaluation were fast, the number of coalitions is astronomically large.

The paper addresses this challenge with approximation methods. The main method used in this project is Truncated Monte Carlo Shapley, which estimates the expectation over random permutations instead of summing over all coalitions exactly.

### B. TMC-Shapley Estimator

For each sampled permutation $\pi$, the algorithm starts with the empty coalition and adds training points one at a time. When point $i$ is added, its marginal contribution in that permutation is:

$$\Delta_i^{(\pi)} = v(S_{\pi,i} \cup \{i\}) - v(S_{\pi,i}), \tag{2}$$

where $S_{\pi,i}$ is the set of points appearing before $i$ in permutation $\pi$. After $T$ sampled permutations, the estimate is:

$$\hat{\phi}_i = \frac{1}{T} \sum_{t=1}^{T} \Delta_i^{(t)}. \tag{3}$$

The implementation also uses the standard truncation idea: if the coalition score becomes sufficiently close to the full-data score, later marginal contributions are skipped or treated as approximately zero. This reduces computation because many late additions no longer change the model much.

### C. Algorithmic Workflow

The practical workflow used by the new extension is:

a. Load the breast-cancer data and create a reproducible train/test split.
b. Standardize features using statistics from the training set only.
c. Corrupt a known subset of training labels, keeping the true indices for evaluation.
d. Compute TMC-Shapley values by sampling random permutations and recording marginal test-accuracy changes.
e. Compute baseline scores: leave-one-out, loss-based suspicion, and random selection.
f. Select the lowest-valued or most suspicious points under a fixed audit budget.

g. Evaluate two interventions: removing selected points and correcting selected labels.
h. Save all results to CSV files and render figures for the report and presentation.

This workflow is important because it keeps the experiment reproducible. Every table in the final report is generated from saved result files rather than manually typed from memory. The report therefore connects the theoretical game, the implementation, and the final evidence.

### D. New Implementation Detail

The original repository imports TensorFlow even for experiments that only require KNN. To make the extension lightweight and reproducible with the packages actually needed for these experiments, I added a new file:

`mywork/shapley_extension_experiments.py`

This file implements the KNN TMC-Shapley pipeline directly using NumPy and scikit-learn. The most important efficiency improvement is precomputing the distance matrix between test points and training points. For a coalition $S$, the algorithm selects the columns of this matrix corresponding to $S$, finds the nearest neighbors, computes predictions, and returns test accuracy. This avoids repeatedly recomputing all distances from scratch.

The new code also exports every major output to CSV and PNG files under:

`mywork/extension_results/`

This makes the final results reproducible and easy to inspect from the notebook, report, and presentation.

### E. Sanity Checks

Several sanity checks were used to avoid reporting unsupported results. First, the Shapley efficiency gap was computed by comparing the sum of estimated point values with $v(N) - v(\emptyset)$. In the final main run, this gap is effectively zero. Second, random selection was included in every major comparison. Since the corruption rate is 20%, random selection should find approximately 20% corrupted points; the observed random precision of 19.34% in the main run is consistent with that expectation. Third, the repeated-seed experiment checks that the results are not driven by one lucky split. Fourth, the source-level experiment is constructed with one intentionally bad source, making it possible to verify whether group-level Shapley ranks that source as harmful.

These checks do not prove that Data Shapley will solve every data-quality problem, but they make the project internally consistent and reduce the chance that the main conclusion is an artifact of a single run.

## IV. Dataset and Experimental Setup

### A. Dataset

The dataset is the UCI Breast Cancer Wisconsin Diagnostic dataset, available through scikit-learn. It contains 569 examples and 30 real-valued features. Each example describes properties of a digitized fine-needle aspirate image of a breast mass. The classification target is binary: malignant or benign.

The project uses an 80/20 train/test split:

- 455 training examples,
- 114 test examples,
- 30 standardized numerical features,
- binary class labels.

The features are standardized in the new experiments because KNN uses Euclidean distance. Without standardization, features with larger numeric scales can dominate the neighbor search.

### B. Controlled Label Corruption

The main corruption type is label flipping. A fixed fraction of training labels is selected, and each selected label is flipped from 0 to 1 or from 1 to 0. In the main experiment, the corruption rate is 20%, so 91 of 455 training labels are corrupted.

This setting has two advantages. First, the corrupted indices are known, so detection performance can be measured directly. Second, label errors are a realistic data-quality problem in high-stakes domains: human annotators may disagree, data entry can be wrong, and clinical outcomes may be recorded incorrectly.

### C. Learning Algorithm and Value Function

The value function $v(S)$ is the test accuracy of a $K$-nearest neighbors classifier trained on coalition $S$, with $K = 5$. KNN is appropriate for this project because it is non-parametric and fast to refit: training mostly consists of storing the coalition examples. Since Data Shapley requires many repeated training evaluations, this makes the experiment computationally feasible.

The empty coalition value $v(\emptyset)$ is the test accuracy obtained by predicting the majority class. This provides a natural baseline for efficiency:

$$\sum_i \hat{\phi}_i \approx v(N) - v(\emptyset). \tag{4}$$

In the final main run, the numerical efficiency gap is approximately $5.55 \times 10^{-17}$, which is effectively zero and confirms that the stored marginal contributions satisfy the Shapley efficiency property up to floating-point precision.

### D. Evaluation Metrics

The project uses several metrics because data valuation has more than one goal. Test accuracy measures whether cleaning improves the final model. Detection precision measures whether the selected suspicious points are truly corrupted. AUROC and average precision measure ranking quality across all possible thresholds rather than only at one fixed budget. Spearman rank correlation is used in the convergence experiment to compare a partial TMC ranking with the full 120-permutation ranking.

TABLE I
EVALUATION METRICS USED IN THE FINAL EXPERIMENTS

| Metric | Meaning in this project |
|---|---|
| Accuracy | Test accuracy after training on corrupted, cleaned, or reduced data. |
| Precision | Fraction of selected audit points that are truly corrupted. |
| Recall | Fraction of all corrupted points found under the audit budget. |
| AUROC | Threshold-independent ability to rank corrupted points below clean points. |
| Average precision | Precision-recall ranking quality for the corrupted class. |
| Spearman $\rho$ | Rank stability of partial TMC estimates against the full run. |

## V. ORIGINAL PROJECT WORK

This section summarizes the earlier work already present in `Shapley_Extension.ipynb` and `final-presentation.tex`.

### A. Original H1: Corrupted Points Receive Negative Values

The original project first tested whether corrupted labels receive lower Shapley values than clean labels. The result is shown in Table II. Clean points had positive average value, while corrupted points had negative average value.

TABLE II
ORIGINAL H1 RESULT: AVERAGE SHAPLEY VALUE BY DATA GROUP

| Data group | Count | Avg. Shapley value |
|---|---|---|
| Clean labels | 364 | $+0.0021$ |
| Corrupted labels | 91 | $-0.0071$ |

This directly supports the theory. Since a flipped label contradicts the feature pattern, it often reduces KNN accuracy when included in a coalition. Its average marginal contribution is therefore negative.

### B. Original H2: Shapley Removal Outperforms Baselines

The original project then removed the bottom 20% of points according to each method. The goal was to test whether removing low-valued points improves test accuracy and whether the removed set actually contains the corrupted labels. Table III reports the original result.

TABLE III
ORIGINAL H2 RESULT: BOTTOM 20% REMOVAL STRATEGY

| Strategy | Acc. | Gain | Detection |
|---|---|---|---|
| No removal | 88.60% | – | – |
| Random | 87.72% | $-0.88$ pp | 21.98% |
| LOO | 96.49% | $+7.89$ pp | 29.67% |
| Shapley | 96.49% | $+7.89$ pp | 82.42% |

The strongest lesson from Table III is that accuracy alone is not enough to judge data valuation. LOO happened to match Shapley's final accuracy in this run, but it detected far fewer corrupted points. Shapley was more faithful to the true data-quality problem because it identified the mislabeled examples directly.

### C. Original H3: Corruption-Rate Stress Test

The third original experiment tested corruption rates from 10% to 40%. The initial expectation was that detection might degrade at high corruption. Instead, detection improved, as shown in Table IV.

TABLE IV
ORIGINAL H3 RESULT: DETECTION ACROSS CORRUPTION RATES

| Corr. | Base | Shapley | Random | LOO |
|---|---|---|---|---|
| 10% | 88.60% | 47.25% | 7.69% | 20.88% |
| 20% | 85.96% | 79.12% | 17.58% | 30.77% |
| 30% | 77.19% | 92.31% | 32.97% | 45.05% |
| 40% | 72.81% | 96.70% | 43.96% | 63.74% |

This is an important finding. When more labels are corrupted, the harmful pattern becomes more consistent across coalitions. The corrupted points are no longer isolated mistakes; they form a stronger negative signal, so Shapley detection becomes easier.

### D. What the Original Work Established

The original work established the core feasibility of the project. It showed that the Data Shapley framework can be applied to a real tabular medical dataset, that the sign and magnitude of Shapley values are meaningful under controlled label corruption, and that Shapley-based rankings can identify corrupted labels much better than random selection or LOO. However, the original experiments were still limited in three ways. First, they focused mostly on removal,

while real data cleaning often prefers correction. Second, convergence and robustness were not yet fully established. Third, the original implementation relied on the repository environment, which was difficult to run directly because of the TensorFlow dependency. These gaps motivate the new extension experiments.

## VI. New Extension Experiments

The new extension work goes beyond the original three hypotheses. It asks whether Data Shapley can support a realistic data-cleaning workflow and whether the results are robust under different evaluation conditions.

### *A. New Main Experiment: Shapley, LOO, Loss, and Random*

The main new experiment uses 20% label corruption, 120 TMC-Shapley permutations, and an audit budget equal to the number of corrupted labels: 91 points. Four selection methods are compared:

- **TMC-Shapley:** select the lowest Shapley values.
- **LOO:** select the lowest leave-one-out values.
- **Loss score:** select examples with highest logistic training loss, represented as lowest negative loss.
- **Random:** select a random subset of the same size.

Table V reports detection and utility. "Precision" means the fraction of selected points that were truly corrupted. "Removal accuracy" trains after deleting selected points. "Relabel accuracy" trains after correcting selected labels back to ground truth.

TABLE V
New Main Experiment: 20% Label Corruption and 91-Point Audit Budget

| Method | Corrupt found | Precision | AUROC | Avg. precision | Removal acc. | Relabel acc. |
|---|---|---|---|---|---|---|
| TMC-Shapley | 71/91 | 78.02% | 0.9500 | 0.8492 | 97.37% | 96.49% |
| LOO | 25/91 | 27.47% | 0.5573 | 0.2472 | 93.86% | 95.61% |
| Loss score | 79/91 | 86.81% | 0.9621 | 0.9255 | 95.61% | 94.74% |
| Random | 17.6/91 | 19.34% | 0.4959 | 0.2003 | 90.22% | 92.32% |

The results show three things. First, Shapley is much better than random and LOO at detecting corrupted labels. Second, loss is a strong corruption detector, which is expected because mislabeled examples often have high loss. Third, Shapley gives the best KNN utility after cleaning: relabeling Shapley-selected points increases accuracy from 90.35% to 96.49%, and removing them increases accuracy to 97.37%.

This result should be interpreted carefully. The loss baseline finds more corrupted labels than Shapley in this particular run, but its relabeling accuracy is lower. This can happen because detection count and downstream utility are related but not identical. Some corrupted points may be easy to identify by loss but may not be the most damaging points for the KNN value function. Shapley is designed to estimate contribution to the chosen value function, so its ranking can be more aligned with improving KNN accuracy even when another method has slightly higher corruption precision.

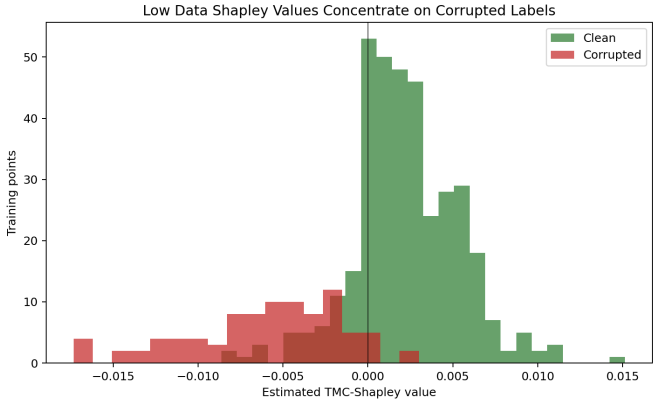


Fig. 1. Distribution of estimated TMC-Shapley values. Corrupted labels are shifted toward lower and negative values.

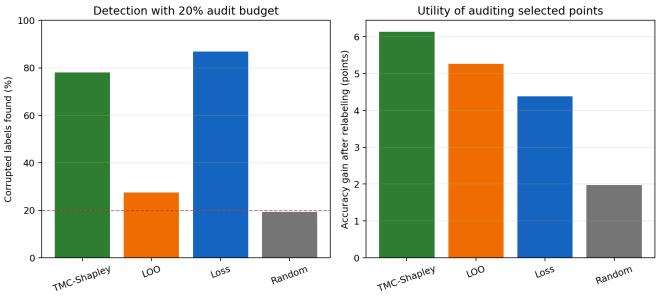


Fig. 2. Main method comparison. Shapley substantially improves corruption detection relative to LOO and random selection, and gives the best KNN relabeling utility in this run.

### *B. Extension A: Budgeted Data Cleaning*

The original project mostly considered removal. The new extension asks a more realistic question: if a human can review only $b$ records, which records should be audited? After auditing, the selected labels are corrected and the model is retrained.

This is more practical than simple deletion. In real datasets, a suspicious record may contain useful feature information even if the label is wrong. Correcting the label preserves the data point while fixing its harmful contribution.

The audit setting also changes the interpretation of the result. A removal experiment asks, "What happens if we throw away suspicious data?" An audit experiment asks, "What happens if a domain expert spends time checking the most suspicious records?" In medical or scientific datasets, the second question is usually more realistic. A patient's measurements should not be discarded if the only problem is a wrong label; the better intervention is to correct the label.

Table VI and Fig. 3 show the budgeted cleaning curve. With no audit, corrupted-label KNN accuracy is 90.35%.

With a 91-label audit selected by Shapley, accuracy reaches 96.49%, while random auditing reaches only 92.28%.

TABLE VI
BUDGETED RELABELING ACCURACY

| Budget | TMC-Shapley | LOO | Loss | Random |
|---|---|---|---|---|
| 0 | 90.35% | 90.35% | 90.35% | 90.35% |
| 22 | 93.86% | 94.74% | 91.23% | 90.91% |
| 45 | 94.74% | 94.74% | 93.86% | 91.51% |
| 68 | 94.74% | 95.61% | 93.86% | 91.58% |
| 91 | 96.49% | 95.61% | 94.74% | 92.28% |

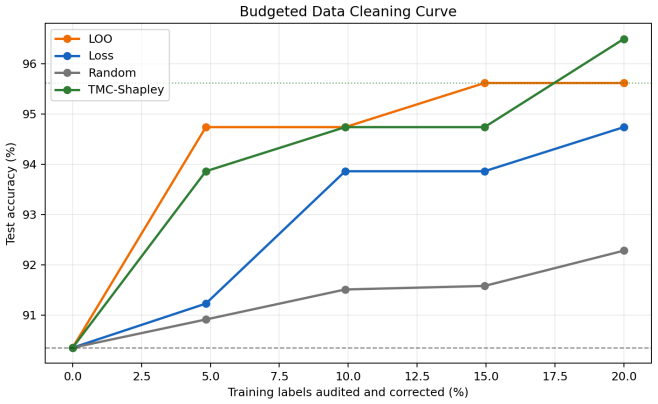


Fig. 3. Budgeted data-cleaning curve. Shapley-guided auditing recovers accuracy much faster than random auditing.

The curve shows why ranking quality matters. At the full 91-point audit budget, random selection improves accuracy from 90.35% to only 92.28%, while Shapley reaches 96.49%. The same amount of human effort therefore produces a much larger accuracy gain when it is guided by a meaningful data-valuation score.

### C. Extension B: Correct Convergence Analysis

The earlier notebook contained a convergence block, but it did not correctly append the marginal contributions returned by each call to `one_iteration`. Therefore, the new implementation stores every TMC marginal vector and computes performance at checkpoints.

Table VII shows that even 5 permutations already produce a detection precision of 67.03%, far above random. More permutations improve AUROC and rank stability. The Spearman correlation with the full 120-permutation ranking rises from 0.5510 at 5 iterations to 0.9516 at 80 iterations.

This result is useful for two reasons. First, it confirms that the final ranking is not arbitrary: as more permutations are averaged, the ranking becomes more stable. Second, it gives a practical scaling insight. If the goal is only to find a rough set of suspicious examples, a small number of permutations may already be useful. If the goal is to report precise values or make high-stakes audit decisions, more permutations are preferable.

TABLE VII
CORRECTED TMC CONVERGENCE METRICS

| Iter. | Precision | AUROC | Spearman $\rho$ |
|---|---|---|---|
| 5 | 67.03% | 0.8499 | 0.5510 |
| 10 | 65.93% | 0.8636 | 0.6373 |
| 20 | 71.43% | 0.8879 | 0.7263 |
| 40 | 69.23% | 0.9243 | 0.8499 |
| 80 | 73.63% | 0.9356 | 0.9516 |
| 120 | 78.02% | 0.9500 | 1.0000 |

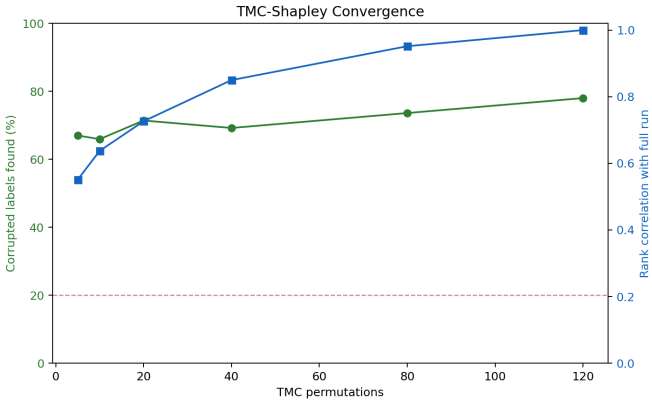


Fig. 4. Convergence of detection precision and rank stability as TMC permutations accumulate.

### D. Extension C: Robustness Across Corruption Types

The next extension tests three corruption types: label flips, feature noise, and outlier shifts. This is important because a method that only works for one artificial corruption setting may not generalize.

The results are shown in Table VIII. Shapley is strongest for label flips: it detects 72.53% of corrupted labels and reaches AUROC 0.9254. It is weaker for feature noise and outlier shifts. This limitation is meaningful. A feature-noisy point can still have the correct label, and KNN may ignore extreme outliers if they are far from test examples. Therefore such points may not consistently create negative marginal contributions.

This should not be interpreted as a failure of the method. It clarifies what the value function is measuring. Since $v(S)$ is test accuracy, Shapley assigns low value to points that reduce test accuracy. A feature-noisy point that does not change many nearest-neighbor votes may not reduce accuracy, even if it looks corrupted to a human. Detecting such points may require a different value function, such as calibration, robustness, or a feature-space anomaly score. The project therefore identifies both a strength and a boundary of Data Shapley: it is highly effective for harmful label noise, but not

every visually abnormal feature vector is harmful under the chosen model and metric.

TABLE VIII
CORRUPTION-TYPE ROBUSTNESS

| Type | Base | After rem. | Prec. | AUROC |
|---|---|---|---|---|
| Label flip | 81.58% | 97.37% | 72.53% | 0.9254 |
| Feature noise | 96.49% | 98.25% | 13.19% | 0.5847 |
| Outlier shift | 97.37% | 98.25% | 10.99% | 0.6217 |

### *E. Extension D: Repeated-Seed Robustness*

To avoid relying on one lucky split, the main pipeline was repeated across five independent train/test splits and corruption draws. Table IX summarizes the results.

TABLE IX
REPEATED-SEED SUMMARY ACROSS FIVE RUNS

| Method | Mean prec. | Std. dev. | Mean AUROC |
|---|---|---|---|
| TMC-Shapley | 69.23% | 4.40% | 0.9037 |
| LOO | 29.23% | 7.56% | 0.5841 |
| Loss score | 87.47% | 2.76% | 0.9745 |
| Random | 19.92% | 1.14% | 0.4995 |

The repeated-seed experiment strengthens the project because it shows that Shapley detection is not a one-run accident. TMC-Shapley remains far above random and LOO across independent splits. Loss remains the strongest pure detector, but it lacks the cooperative-game fairness interpretation and does not directly allocate model value among data points.

The standard deviation is also informative. Shapley precision varies by 4.40 percentage points across the five runs, while LOO varies by 7.56 percentage points. This suggests that LOO is not only weaker on average but also less stable. A data-cleaning method used in practice should be reliable across different train/test splits, not only successful in one favorable split.

### *F. Extension E: Model-Transfer Validation*

Since the value function uses KNN, Shapley values are optimized for KNN accuracy. A natural question is whether the selected corrections also help other downstream models. Table X reports results for KNN, logistic regression, and random forest.

TABLE X
MODEL-TRANSFER VALIDATION

| Model | Method | Base | Fixed | Gain |
|---|---|---|---|---|
| KNN | TMC-Shapley | 90.35% | 96.49% | +6.14 pp |
| KNN | LOO | 90.35% | 95.61% | +5.26 pp |
| KNN | Loss | 90.35% | 94.74% | +4.39 pp |
| Logistic | TMC-Shapley | 92.11% | 94.74% | +2.63 pp |
| Logistic | Loss | 92.11% | 96.49% | +4.39 pp |
| Random forest | TMC-Shapley | 96.49% | 96.49% | +0.00 pp |

The result is nuanced. Shapley-selected corrections transfer positively to logistic regression, but the largest gain is for KNN, which is expected because KNN defines the valuation game. Random forest already performs very well under this corruption draw, so there is little room for improvement.

This experiment is important because it prevents an overclaim. The project does not prove that one set of Shapley values is universally optimal for every model. Instead, it shows a more precise and defensible conclusion: Shapley values are tied to the learning algorithm and performance metric used in the game, but cleaning points selected by the KNN game can still help another model when the selected records are genuinely mislabeled. This is why the model-transfer result is presented as validation, not as a universal guarantee.

### *G. Extension F: Source-Level Shapley*

The final extension changes who the players are. Instead of treating individual records as players, I group records into ten sources, representing possible data providers. One source is intentionally made much noisier than the others.

Table XI and Fig. 5 show that source 3 has 30 corrupted records out of 46 and receives the lowest source-level Shapley value, $-0.1043$. All other sources have only two corrupted examples and positive values.

TABLE XI
SOURCE-LEVEL SHAPLEY: LOWEST-VALUED PROVIDERS

| Source | Size | Corrupt | Value |
|---|---|---|---|
| 3 | 46 | 30 | $-0.1043$ |
| 8 | 45 | 2 | 0.0264 |
| 2 | 46 | 2 | 0.0303 |
| 1 | 46 | 2 | 0.0367 |
| 4 | 46 | 2 | 0.0409 |

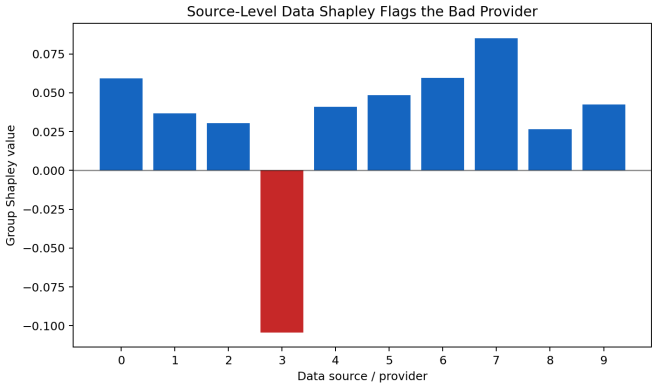


Fig. 5. Source-level Data Shapley correctly ranks the intentionally bad provider as the lowest-valued source.

This is a significant conceptual extension. In many real projects, data arrives from multiple hospitals, annotators, devices, or vendors. Source-level Shapley can identify not only individual bad records, but also unreliable providers.

The provider-level experiment also connects directly to the efficiency and accountability motivations behind Shapley value. If a model is trained using data from multiple sources, one may want to reward high-quality sources or investigate low-quality sources. A simple average error rate may miss how each source interacts with the rest of the training data. Group Shapley keeps the coalition logic but changes the player definition: now a player is a source rather than a single row. The fact that the intentionally bad source receives the lowest group value shows that the framework can scale conceptually from record-level cleaning to provider-level evaluation.

## VII. Discussion

### A. What the Results Show

The original results and new extension results tell a consistent story. Corrupted labels receive low or negative Shapley values because they reduce model accuracy when added to many coalitions. Removing or correcting low-Shapley points improves accuracy because those points are genuinely harmful to the learned classifier. Random selection behaves approximately as expected: under 20% corruption, it finds about 20% corrupted points. LOO is better than random in some accuracy outcomes but is not a reliable detector because it ignores the many coalition contexts captured by Shapley.

The strongest practical result is the audit-budget experiment. In real data cleaning, the problem is often not "can I remove bad data?" but "which examples should I spend time checking?" Shapley provides a principled answer. Under a 91-point audit budget, TMC-Shapley detects 71 corrupted labels and raises KNN accuracy from 90.35% to 96.49% after correction.

### B. Why Loss Can Beat Shapley in Detection

The loss baseline detects 79 of 91 corrupted labels in the main run, which is higher than Shapley's 71. This does not invalidate the Shapley contribution. Loss and Shapley answer different questions. Loss asks whether the current fitted model finds an example surprising. Shapley asks how much the example contributes to the learning algorithm's value across coalitions. Loss is therefore a strong diagnostic heuristic, while Shapley is a fair value allocation rule. The project becomes stronger by including both: it shows that Shapley is competitive with a strong machine-learning baseline while retaining the cooperative-game interpretation.

### C. Why Feature Noise Is Harder

The corruption-type experiment shows that label flips are much easier to detect than feature noise or outlier shifts. This is reasonable. A wrong label directly contradicts the classification objective. A noisy feature vector, however, may still have the correct label, and if it lies far from the decision boundary it may not affect test predictions much. In Shapley terms, such a point is not necessarily harmful across coalitions, so it may not receive a strongly negative value.

### D. What Makes the Extension Substantial

The extension is substantial because it adds new experimental questions rather than only rerunning the original paper. The original paper proposes Data Shapley as a valuation principle, and the earlier project demonstrated that principle on a controlled corruption task. The added work changes the evaluation from a single demonstration into a broader investigation. It asks how many corrupted points are found, how much accuracy is recovered, how stable the ranking is as TMC samples increase, whether the result survives repeated splits, whether cleaning transfers to other models, and whether the same framework can identify bad data sources.

These additions make the project more complete from both a cooperative-game perspective and a machine-learning perspective. From the game-theory perspective, the project now tests individual and grouped players, verifies efficiency numerically, and compares Shapley with methods that do not satisfy the same axioms. From the machine-learning perspective, it evaluates the operational consequence of valuation: improved test accuracy after targeted data correction.

### E. Defensible Claims

Based on the evidence, the strongest defensible claims are:

- Data Shapley assigns lower values to corrupted labels in the controlled breast-cancer experiment.
- Shapley-based selection is much stronger than random selection and LOO for corrupted-label detection.

- Under a fixed audit budget, correcting Shapley-selected labels substantially improves KNN accuracy.
- The Shapley ranking becomes more stable as more TMC permutations are used.
- The method is strongest for label corruption and weaker for feature noise under the chosen KNN accuracy value function.
- Source-level Shapley can identify an intentionally bad data provider in the grouped-source extension.

The report does not claim that Shapley always beats every possible detector. In fact, the loss baseline has higher corrupted-label precision in the main new run. The more accurate claim is that Shapley gives a principled cooperative-game valuation and produces strong practical cleaning performance, especially compared with random and LOO baselines.

## VIII. Threats to Validity

The conclusions are supported by controlled experiments, but it is important to state what could affect their interpretation.

### A. Internal Validity

The corruption indices are generated synthetically, so the project can measure detection precision exactly. This is a strength for evaluation, but it also means the corruption mechanism is simpler than real annotation errors. To reduce internal-validity concerns, the report compares Shapley with random selection, LOO, and loss-based scoring, and it verifies that the random baseline behaves as expected under 20% corruption.

### B. External Validity

The dataset is a real medical benchmark, but it is still a small tabular dataset with 569 examples. Results on larger image, text, or multi-institution datasets may differ. The source-level experiment is therefore best interpreted as a proof of concept for provider accountability, not as a claim that the same numerical values would hold for every data-sharing setting.

### C. Construct Validity

The value function in this project is KNN test accuracy. This is a clear and measurable utility, but it is not the only possible definition of data value. If the objective were AUROC, calibration, fairness, or robustness, the Shapley values could change. Therefore, the strongest claim is model-and-metric specific: the reported values measure contribution to the chosen KNN accuracy game.

## IX. Limitations

The project has several limitations:

- **Computation:** TMC-Shapley requires repeated coalition evaluation. Even with KNN and precomputed distances, larger datasets would require more optimization.
- **Model dependence:** Values are tied to the selected learning algorithm and metric. A point valuable for KNN may not be equally valuable for a neural network.
- **Dataset size:** The breast-cancer dataset is useful and interpretable, but it is small. Larger datasets would better test scalability.
- **Synthetic corruption:** Label flips are controlled and measurable, but real data errors can be more complex.
- **Feature corruption:** The method is weaker for feature noise and outlier shifts, indicating that additional diagnostics may be needed.

These limitations do not weaken the main contribution; instead, they clarify where Data Shapley is strongest and where future work is needed.

## X. Future Work

Several natural extensions remain. First, the project could compare sampled TMC-Shapley values with exact or semi-exact KNN Shapley formulas on smaller subsets. This would measure approximation error directly. Second, the audit workflow could be extended to active learning: rather than auditing only the lowest-valued points, one could combine low Shapley value with prediction uncertainty or clinical risk. Third, the source-level experiment could be tested on real multi-institution data, where different hospitals or annotators contribute different parts of the dataset. Fourth, future experiments could evaluate other value functions, such as F1 score, AUROC, calibration error, or fairness metrics. Changing the value function would answer different data-quality questions.

Another important direction is uncertainty quantification. A single Shapley estimate is useful, but a confidence interval around each value would be more appropriate for high-stakes review. Points with very negative values and tight confidence intervals would be strong audit candidates, while points with uncertain values might require more sampling before a decision is made.

## XI. Reproducibility

The final project files are organized as follows:

- `final-project-report.tex`: this IEEE-style report source.
- `final-project-report.pdf`: compiled report for submission.
- `final-presentation.tex`: earlier presentation based on the original work.

- `final-final-presentation.tex`: final combined presentation source.
- `mywork/Shapley_Extension.ipynb`: final notebook that loads and explains the experiment artifacts.
- `mywork/shapley_extension_experiments.py`: reproducible experiment code.
- `mywork/extension_results/`: generated CSV tables and PNG figures.

The new experiments can be regenerated with:

```
python
mywork/shapley_extension_experiments.py
```

The report is written for the standard IEEE conference class and can be compiled in Overleaf or a local TeX installation that includes `IEEEtran`. The numerical tables are traceable to the CSV files in `mywork/extension_results/`, and the figures used in the report are stored in `mywork/extension_results/figures/`.

## XII. Conclusion

This project applies cooperative game theory to the practical problem of bad training-data detection. The original work showed that Data Shapley identifies corrupted labels on the UCI Breast Cancer Wisconsin Diagnostic dataset: corrupted points receive negative values, Shapley-based removal improves accuracy, and detection remains strong across corruption rates. The new extension makes the project substantially stronger by adding a reproducible implementation and evaluating Data Shapley as a practical data-cleaning tool.

The most important final result is that TMC-Shapley found 71 of 91 corrupted labels under a 20% audit budget and improved KNN accuracy from 90.35% to 96.49% after correcting selected labels. The repeated-seed experiment confirms that this effect is robust beyond one split, and the source-level experiment shows that the same Shapley idea can identify unreliable data providers.

Overall, the project demonstrates that the Shapley value is not only a theoretical fairness concept from cooperative games. It can be operationalized as a data-quality score that detects harmful examples, supports limited manual auditing, and extends naturally from individual records to data sources. The strongest aspect of the project is this connection between theory and practice: the same marginal-contribution logic that defines fair payoff allocation in a cooperative game also tells a machine-learning practitioner which training labels are most worth checking. That makes Data Shapley a useful bridge between cooperative game theory and responsible data-centric machine learning.

## References


[1] A. Ghorbani and J. Zou, "Data Shapley: Equitable valuation of data for machine learning," in *Proc. 36th International Conference on Machine Learning (ICML)*, 2019, pp. 2242–2251.

[2] L. S. Shapley, "A value for n-person games," in *Contributions to the Theory of Games II*, H. W. Kuhn and A. W. Tucker, Eds. Princeton, NJ, USA: Princeton University Press, 1953, pp. 307–317.

[3] W. H. Wolberg, W. N. Street, and O. L. Mangasarian, "Breast cancer Wisconsin (diagnostic) data set," UCI Machine Learning Repository, 1995.

[4] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.