Efficiently Mitigating the Impact of Data Drift on Machine Learning Pipelines

Sijie Dong
Université Paris Cité
Paris, France
sijie.dong@etu.u-paris.fr

Qitong Wang
Université Paris Cité
Paris, France
qitong.wang@etu.u-paris.fr

Soror Sahri
Université Paris Cité
Paris, France
soror.sahri@parisdescartes.fr

Themis Palpanas
Université Paris Cité
Paris, France
themis@mi.parisdescartes.fr

Divesh Srivastava
AT&T
Bedminster, NJ, USA
divesh@research.att.com

ABSTRACT

Despite the increasing success of Machine Learning (ML) techniques in real-world applications, their maintenance over time remains challenging. In particular, the prediction accuracy of deployed ML models can suffer due to significant changes between training and serving data over time, known as *data drift*. Traditional data drift solutions primarily focus on detecting drift, and then retraining the ML models, but do not discern whether the detected drift is harmful to model performance. In this paper, we observe that not all data drifts lead to degradation in prediction accuracy. We then introduce a novel approach for identifying portions of data distributions in serving data where drift can be potentially harmful to model performance, which we term Data Distributions with Low Accuracy (DDLA). Our approach, using decision trees, precisely pinpoints low-accuracy zones within ML models, especially Black-box models. By focusing on these DDLAs, we effectively assess the impact of data drift on model performance and make informed decisions in the ML pipeline. In contrast to existing data drift techniques, we advocate for model retraining only in cases of harmful drifts that detrimentally affect model performance. Through extensive experimental evaluations on various datasets and models, our findings demonstrate that our approach significantly improves cost-efficiency over baselines, while achieving comparable accuracy.

PVLDB Reference Format:
Sijie Dong, Qitong Wang, Soror Sahri, Themis Palpanas, and Divesh Srivastava. Efficiently Mitigating the Impact of Data Drift on Machine Learning Pipelines. PVLDB, 17(11): 3072 - 3081, 2024.
doi:10.14778/3681954.3681984

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/SiSijie/data-drift-in-ML.

Proceedings of the VLDB Endowment, Vol. 17, No. 11 ISSN 2150-8097.
doi:10.14778/3681954.3681984

1 INTRODUCTION

Background. One of the most important quality issues of ML systems is data drift [4, 36], also known as data shift [23, 50]. Drift often occurs in two ways: covariate shift [23], which refers to changes in the distribution of input features between training and serving data; and concept drift [19, 37, 50, 51], which occurs when the relationship between input features and the target variable changes over time. In this paper, we focus on covariate shifts in our data drift problem. Data drift can be caused by various factors, such as changes in the environment, user behaviors, or system upgrades [19].

If left unaddressed, data drift can adversely impact the performance of ML models, leading to incorrect predictions and suboptimal decision-making. To address the issue of data drift, a common approach is periodically retraining the ML models on the new data to adapt to the data distribution changes. However, this can be computationally expensive. As a result, there is a pressing need to develop efficient and effective methods to mitigate the impact of data drift on the performance of ML models.

Motivation. To motivate our work, let us consider the application of an ML pipeline developed using the UCI Adult dataset [8]. This pipeline predicts if an individual's annual income exceeds \$50k, using attributes like *race*, *gender*, *age*, *workclass*, *education*, and *hours worked per week*. Figure 1 presents a hypothetical data distribution based on the attributes *education* denoting the highest level of education achieved and *workclass* indicating the employer type, to motivate our study. These distributions are for simplicity of exposition, acknowledging that the dataset's distributions are substantially more complex than the simplified ones presented here.

We observe that the proportion of *Doctorates working for Private employers (DP)* and *Doctorates working for Federal-Government employers (DG)* is much smaller than the proportion of Bachelors working for Private employers (BP), Masters working for Private employers (MP), and Masters working for Federal-Government employers (MG). If a model is trained using this data, it may result in inaccurate predictions for the income level of *DP* and *DG*. This could have real-world implications, as some organizations may use these predictions, e.g., for career counseling.

Example: Consider the data drift caused by training and serving data originating from different cities that have different industries. The educational backgrounds and industrial compositions vary among people from different cities. In Scenario 1 of Fig. 1, there's

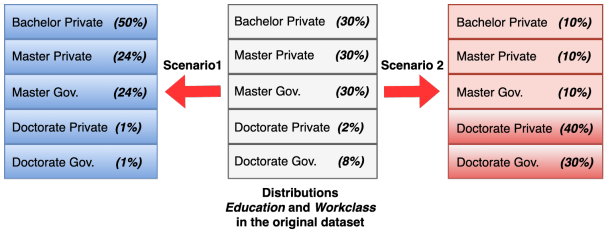


Figure 1: Two different scenarios of data drift, after adding several batches of serving data.

a minor decrease in *MP, MG, DP* and *DG* data, with a significant increase in *BP*. Since *BP* is already well-represented in training data, this drift doesn't substantially impact model accuracy, making retraining unnecessary. In Scenario 2 of Fig. 1, we assume a substantial increase in the proportion of *DP* and *DG* in the data. This drift in the data distribution could lead to lower prediction accuracy with the model trained on the original dataset. Therefore, it becomes crucial to retrain the model to accommodate this change.

A straightforward method to identify areas of low accuracy would be to compute the model's accuracy for all possible combinations of attributes (using, e.g., data cubes [28]). Fig. 1 shows a dataset with two attributes: *education*, *workclass*, resulting in attribute combinations (BP, MP, MG, DP, DG, etc.). The model's accuracy for each combination needs assessment. However, the Adult dataset actually comprises 14 attributes, leading to a vast number of potential attribute combinations. This method quickly becomes computationally unfeasible as the number of attributes increases, especially for large attribute datasets like time series or images, due to the exponential increase in combinations. Thus, a method to efficiently detect distributions with low prediction accuracy is essential to swiftly detect harmful data drifts similar to those observed in Scenario 2 and subsequently retrain the model. However, in Scenario 1, we do not retrain the model. Compared to other methods of data drift detection and model retraining, we aim to maintain comparable accuracy of model prediction while effectively reducing the costs associated with retraining.

Challenges. The examples illustrate three key challenges. First, we need to identify data distributions where the model's prediction accuracy drops significantly. Black-box models do not provide insight into why they make specific predictions, making this challenge more difficult in analyzing their predictions' accuracy on specific distributions directly. Second, we need to distinguish harmful data drift from benign ones: in Scenario 2 of the example above, only low-accuracy predictions require retraining. Third, acquiring additional labeled samples for model retraining is costly, and we need to select more valuable samples to improve the retraining efficiency.

Our solution. To address these challenges, we propose a novel approach that analyses data distributions of a black-box model and identifies those that may cause an accuracy drop in the model. The underlying intuition is that in real-world settings where future data drift is unpredictable, such distributions can determine the potentially harmful data drift, and then decide when to retrain. Our approach involves three main steps (see Figure 2).

Identification of Data Distributions with Low Accuracy (DDLAs). To address the first challenge, we use decision trees in a novel way to specifically identify portions of data distributions where the model's prediction accuracy is notably lower than its overall performance. Unlike many existing works [17, 25, 27, 31, 56], we do not use decision trees to explain the behavior of the trained black-box model. Instead, our approach generates a new type of training data by re-labeling the predictions of the black-box model. Additionally, by analyzing the structure of the decision tree, we can partition the data and estimate the prediction accuracy for each portion of the data distribution, thereby providing a more targeted and efficient way of addressing prediction inaccuracies.

Harmful Data Drift Detection. Using the DDLAs identified, we can detect changes in DDLAs' ratio between training and serving data and categorize them into harmful and benign data drift. Accordingly, we decide whether to perform the model's retraining or not. This step allows us to mitigate unnecessary retraining in ML pipelines.

Active Learning for model retraining. We leverage active learning to improve retraining efficiency. Specifically, we gather samples from only DDLAs within the serving data that exhibit harmful data drift. Once annotated by domain experts, such data is used for retraining the black-box model and enhancing its performance.

Contributions. We summarize our contributions as follows. First, we propose an approach to predict potential data that impacts the accuracy of ML black-box models. In contrast to existing solutions, we retrain models only in cases of *harmful* data drifts, leading to scalable ML pipelines in real settings. We use decision trees to specifically identify regions within the ML model, where accuracy is low. Second, we employ active learning techniques to enhance the efficiency of the retraining process, when such retraining is deemed necessary. Finally, we experimentally evaluate our approach on various datasets, using different ML models, to demonstrate that our approach maintains comparable accuracy while effectively reducing the costs associated with retraining compared to baselines.

2 RELATED WORK

Addressing quality issues in ML pipelines has grown in interest in recent years. Several works studied the relationship between data quality and ML-model performance, including the impact of data quality on ML [10, 16, 24, 29, 45], its link to MLOps [42, 53], cleaning for ML [30, 32] and quality validation of ML pipelines [44].

Much work has investigated the effect of data drift on ML performance. Many solutions for evaluating ML models' performance under data drift are devoted to drift detection methods for concept shifts [3, 4, 7, 19, 33, 55] or covariate shift [21, 39]. Data drift detection methods can be categorized into several distinct classes, notably: statistics-based detection methods such as Kolmogorov-Smirnov (KS) test [15], Maximum Mean Discrepancy (MMD) [12, 22] and Mahalanobis distance (MD) [41]; and ML-based methods such as Classifier Two-Sample Tests (C2ST) [35] and MMD-D [34].

Other prior work about data drift compared different drift detection methods [18] and presented drift detection methods for streaming data [49]. However, most of the proposed solutions rely on periodic retraining, which may result in performance degradation and cost increase [15]. In contrast to prior work, we mitigate periodic retraining by proposing a method to detect harmful data

Table 1: Data Distribution Before and After Drift ("-" denotes High or Low values, with cases 1 and 2 covering all combinations except for cases 3 and 4)

Case	Before drift (training data)		After drift (serving data)		Harmful Drift
	Acc.	Sample Size	Acc.	Sample Size	
1	-	-	High	-	No
2	-	-	-	Low	No
3	High	Low	Low	High	Yes
4	Low	Low	Low	High	Yes

drift through areas with low model prediction accuracy, in combination with active learning techniques to retrain models. Similar efforts have been made by Matchmaker [36] and Detectron [21]. Matchmaker proposed to mitigate retraining for the data drift problem, but it cannot identify harmful drift, leading to potential unnecessary costs. Detectron can detect harmful data drifts, but lacks interpretability, as it does not provide insights into the specific regions within the data where the harmful drift originates. Moreover, Detectron has a rather high computational cost, and is limited to scenarios with small batch sizes in the serving data.

Our work also differs from the interpretability of black-box models [14, 17, 25, 27, 31, 43, 45, 56]. In contrast to prior works that interpret the predictions using decision trees [17, 27, 43, 56], our approach leverages decision trees as a lightweight diagnostic tool to systematically identify the harmful areas of the data where prediction accuracy drops. We target these low-accuracy areas within the model's predictions for our active learning (AL) strategy. AL aims to achieve high prediction accuracy with low labeling cost, by querying the most informative and representative points from unlabeled data. Prior AL strategies are divided into two main types: information-based and representative-based [6, 40, 46, 48]. Among the information-based strategies, uncertainty sampling, which suggests labeling samples that are the most uncertain, is one of the most popular strategies for AL. As for representative-based strategies, the density-based approach is commonly used, selecting data points based on their density in the feature space to capture informative samples. Our AL strategy differs from prior AL strategies by querying samples from the low-accuracy regions for expert labeling and subsequent retraining upon detecting harmful data drift.

3 PROBLEM DEFINITION

Data drift, is defined as a change in the distribution of the input variables X while the conditional distribution of the target variable Y given X remains the same. It can be expressed as: $\mathcal{P}_t(X) \neq \mathcal{P}_{t+1}(X)$ and $\mathcal{P}(Y \mid X)$ is invariant , where: $\mathcal{P}$ denotes the probability distribution, X. If this change negatively impacts the predictive accuracy of a trained ML model, we refer to it as *harmful data drift*.

Data Distributions with Low Accuracy (DDLAs). Given a black-box model M, which we assume has been well-trained to get high overall accuracy, we define a *DDLA* as a subset of the feature space where the accuracy of M on *DDLA* is lower than the overall accuracy of M. Formally, a *DDLA* is a region that $Acc(M, DDLA) < acc_M$, where acc_M is the overall accuracy of M.

Harmful data drift detection and mitigation. Our goal is to first determine the changes in data distributions of serving data affecting the model's performance and then mitigate their impact. Specifically, we focus on cases where the proportion of DDLAs increases in serving data, indicating harmful data drift. Table 1 depicts the different cases of data distributions before and after a data drift occurs, based on their sample size and the related accuracy. As the model is well-trained, distributions with high sample sizes (HS) generally lead to a high level of accuracy (HA), we then focus on cases of low sample size (cases 3 and 4 in Table 1) which may cause harmful data drift. Case 3 corresponds to distributions changing from HA and low sample size (LS), to low accuracy levels (LA) and HS. Although the model achieves HA in these distributions of training data, the LS suggests that this accuracy might not generalize to broader datasets. Changes in the sample size could potentially transform these regions into DDLAs. Case 4 corresponds to DDLA regions, which may cause harmful data drift when it becomes LA and HS in the serving data.

Notice that our method assumes that the learning ability of the model for each sub-dataset is comparable, we do not take into account the highly difficult-to-learn sub-datasets, because there is no effective way to make the model learn well for them.

4 OUR APPROACH

At the core of our approach, we begin with a black box ML model, for which we possess the training data. In order to detect all cases of harmful data drift, as presented in Table 1, we first update the labeled data by performing a uniform sampling on serving data (Step 1-a). Then, we identify DDLAs of the black box model by leveraging decision trees (Step 1-b). Next, we quantify and compare the proportions of the identified DDLAs between training data and serving data (Step 2). The resulting changes from this comparison allow us to detect harmful data drift and decide the subsequent steps in the ML pipeline, i.e., retraining or not. Only in the case of harmful data drift, we perform active learning for annotating samples from DDLAs by experts, and then use just the instances related to wrong predictions for model retraining (Step 3). Following these steps, our approach encompasses two methods which we refer to as DDLA-Full and DDLA-Opt. DDLA-Full includes all the steps and enables the detection of all cases of harmful drift (cases 3 and 4 in Table 1), as well as the case that new distributions arise in the serving data, for example, training data lacks "High School" value in "Education" attribute, but it appears in serving data. DDLA-Opt, on the other hand, focuses on using only existing labeled data to detect harmful drift, which is related to case 4 in Table 1.

Figure 2 provides a schematic overview of our proposed approach. We next describe the steps of our approach followed by how each method integrates these steps.

4.1 Update labeled data (Step 1-a)

In this step, we use the serving data to update labeled data and allow us to identify DDLAs related to two scenarios: (i) New distributions arise in the serving data resulting in a lack of available information about those new distributions when using existing labeled data to identify DDLAs. The uniform sampling from the serving data will provide the decision tree's training data with new distributional information, which helps mitigate the impact of data drift in such a scenario. (ii) Case 3 in Table 1 which is potential DDLAs that may

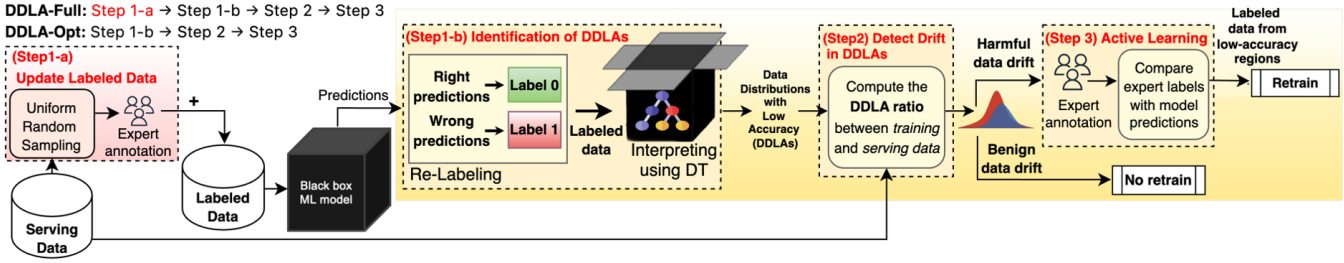


Figure 2: Overview of our approach with its two methods DDLA-Full and DDLA-Opt.

Algorithm 1 Identifying DDLAs

Input: Training dataset D_{train}, Test dataset D_{test}
Output: A set of Data Distributions with low accuracy $DDLA$
Algorithm:

1: $M \leftarrow \text{TrainModel}(D_{train})$, $acc_M \leftarrow \text{Acc.}(M, D_{test})$
2: $Y_{pred} \leftarrow \text{Predict}(M, D_{test})$
3: $C = \{x \in D_{test} | M(x) = y_{D_{test}}\}$, $E = \{x \in D_{test} | M(x) \neq y_{D_{test}}\}$
4: Re-label instances in C and E to create:
$D_{analytic} = \{(x, y_{relabeled}) | x \in (C \cup E)\}$
5: $DT \leftarrow \text{TrainDecisionTree}(D_{analytic})$
6: $N_i \leftarrow \text{IdentifyLeafNodes}(DT)$ for $i = 1$ to n
7: $D_{\text{test}_i} \leftarrow \text{Classify}(DT, D_{test})$ for $i = 1$ to n
8: $Y_{pred} \leftarrow \text{Predict}(M, D_{\text{test}_i})$ for $i = 1$ to n
9: $acc_{D_{\text{test}_i}} \leftarrow \text{Accuracy}(M, Y_{pred}, D_{\text{test}_i})$ for $i = 1$ to n
10: $DDLA \leftarrow \emptyset$
11: **for** $i = 1$ to n **do**
12: **if** $acc_{D_{\text{test}_i}} < acc_M$ **then**
13: $DDLA \leftarrow DDLA \cup \{N_i\}$
14: **end if**
15: **end for**
16: **return** $DDLA$

come from the high accuracy and low sample size regions in the original training data. By increasing the samples from serving data, we can enhance the certainty of calculated accuracy for previously high accuracy and low sample size distributions, thereby mitigating this scenario. To identify DDLAs including those related to these two scenarios, we perform uniform random sampling on serving data and then do expert labeling.

Uniform Random Sampling. We input each batch of serving data and output a uniformly randomly selected subset, which can provide sufficient information on the serving data.

Expert annotation. This step takes a selected subset of samples as input and produces a dataset with expert-provided labels as output.

Once labeled, we add them to the existing labeled data.

4.2 Identification of DDLAs using Decision Trees (Step 1-b)

The steps of identifying DDLAs are shown in Algorithm 1. We assume that we are provided with a black-box model M which has been trained on a dataset D_{train}, and we also have a test dataset D_{test}, We then apply the model M to D_{test} to get predictions denoted as $Y_{pred} = \text{Predict}(M, D_{test})$ (line 2). To identify DDLAs, we first proceed to re-label Y_{pred} predictions, and then use the re-labeled data to train a decision tree.

Re-labelling of model's predictions. We introduce an innovative method to identify low-accuracy regions in a black-box model's predictions. Traditional methods that train a decision tree model with the intention of simulating the decision-making process of the black-box model, often fail to maintain consistency in identifying low-accuracy regions. This is because even if the overall accuracy of the decision tree model and the black-box model is similar, it does not guarantee that the two models will have the same low-accuracy regions across different data distributions. To address this issue, we propose a relabeling process where data instances are labeled based on the accuracy of the black-box model's predictions.

We categorize the model predictions Y_{pred} by comparing them with the true labels $y_{D_{test}}$. This categorization splits the predictions into correct predictions, denoted by $C = x \in D_{test} | M(x) = y_{D_{test}}$, and erroneous ones, denoted by $E = x \in D_{test} | M(x) \neq y_{D_{test}}$.

Subsequently, we re-label these categories with new labels: 0 for correct predictions ($x \in C$) and 1 for incorrect ones ($x \in E$), resulting in the set $y_{relabeled}$. This relabeling process forms a new dataset, denoted as $D_{analytic}$, which is defined by the formula: $D_{analytic} = \{(x, y_{relabeled}) | x \in (C \cup E)\}$.

Identifying DDLAs using Decision Trees. We train a decision tree model DT on the generated training set $D_{analytic}$ to identify DDLAs within the black-box model M. Decision trees provide a hierarchical structure of decisions based on different feature values, which allows us to trace the path of decisions and understand the reasoning behind each prediction [43]. Each decision tree leaf node corresponds to a data subset, representing specific feature value combinations. This aspect is crucial for pinpointing underperforming regions in black-box models. In our approach, the decision tree model trained on relabeled data, marks each leaf node as a distinct data distribution, with labels '0' for mostly correct and '1' for mostly incorrect predictions by model M. This way, DDLAs are mostly represented in leaf nodes corresponding to label 1.

In general, trees of greater depth generally yield improved accuracy. However, deeper trees offer more specific features. thus potentially reducing their utility. Striking a balance between specificity and generalizability necessitates meticulous adjustment of the decision tree's maximum depth. Here, parameters like *max depth* and *min samples leaf* play crucial roles. Each time we train a new decision tree to identify DDLAs, we consider optimizing its parameters by tuning *max depth* and *min samples leaf*. We use grid search combined with 10-fold cross-validation to test combinations, ranging from 3 to 10 for *max depth* and 0% to 5% of our dataset for *min samples leaf*. After training the decision tree DT on $D_{analytic}$, we analyze its structure to identify unique data distributions, each represented by a leaf node N_i. We then categorize

instances from the test data D_{test} into these distributions, forming subsets $D_{\text{test}_i} = \{x \in D_{\text{test}} | DT(x) = N_i\}$ for each node N_i.

For each subset D_{test_i}, we calculate M's prediction accuracy as $acc_{D_{\text{test}_i}}$. Our approach identifies DDLAs as distributions with an accuracy below the model's overall accuracy, acc_M. Specifically, $DDLA = N_i | acc_{D_{\text{test}_i}} < acc_M$, and we define D_{DDLA} as the collection of instances in these distributions. After retraining the model, it is necessary to re-identify and update DDLAs. These DDLAs reveal interpretable distributions, emphasizing critical features. The practical utility of DDLAs lies in their ability to explicitly reveal specific flaws within a model, enabling targeted improvements.

4.3 Detection of Harmful Drift from Identified DDLAs in Serving Data (Step 2)

Once the DDLAs are identified, we investigate their proportions in the serving data D_{serving}. We distinguish two possible scenarios, thereby determining the subsequent steps in the ML pipeline. The key aspect of our analysis is the comparison of the ratio of DDLAs between the training data D_{train} and the serving data D_{serving}. Any change in the ratio of DDLAs corresponds to a data drift. The ratios are defined as follows: $\text{Ratio}_{\text{train}} = \frac{|D_{DDLA} \cap D_{\text{train}}|}{|D_{\text{train}}|}$ for the training data and $\text{Ratio}_{\text{serving}} = \frac{|D_{DDLA} \cap D_{\text{serving}}|}{|D_{\text{serving}}|}$ for the serving data.

Depending on whether $\text{Ratio}_{\text{serving}}$ is greater than, less than, or equal to $\text{Ratio}_{\text{train}}$, we have two distinct scenarios:

- *Case 1: Decrease or No Change in DDLA Ratio.* If $\text{Ratio}_{\text{serving}} \leq \text{Ratio}_{\text{train}}$, the proportion of DDLAs in the serving data is then not higher than in the training data. Consequently, no additional action is required. We term such cases as *benign data drift*.
- *Case 2: Increase in DDLA Ratio.* If $\text{Ratio}_{\text{serving}} > \text{Ratio}_{\text{train}}$, we discern an increase in the proportion of DDLAs in the serving data. Such cases, termed as *harmful data drift*, require the implementation of active learning to label additional samples in the DDLAs, which we present in the next step.

An increase in the DDLA ratio doesn't always necessitate model retraining. To avoid unnecessary retraining for minor increases or low DDLA ratios, we set thresholds θ_{DDLA} and θ_{inc}. Retraining is essential if: $\frac{\text{Ratio}_{\text{serving}} - \text{Ratio}_{\text{train}}}{\text{Ratio}_{\text{train}}} > \theta_{\text{inc}} \wedge \text{Ratio}_{\text{serving}} > \theta_{\text{DDLA}}$. For instance, a minor DDLA ratio increase from 0.1% to 0.2% might not need retraining. Properly setting θ_{inc} and θ_{DDLA} balances performance with retraining costs. We provide a strategy for setting θ_{inc} and θ_{DDLA}. Based on experimental findings indicating negligible effects of θ_{inc} variations on the outcomes, we set θ_{inc} to 0.5 to ensure stable performance. Regarding θ_{DDLA}, it significantly impacts the need for model retraining, as revealed by our experiments. Thus, we have adopted a dynamic setting for θ_{DDLA}, focusing on identifying DDLAs whose ratios increase in the serving data. We determine a specific ratio where the increase of these DDLAs on the serving data corresponds to a 10% decrease in the model's prediction performance, setting it as the new threshold for θ_{DDLA}.

4.4 Enhancing Model Performance with Active Learning (Step 3)

The steps of active learning are shown in Algorithm 2. In cases of harmful data drift, retraining the model becomes essential. Our approach employs a new active learning strategy to efficiently use limited labeled samples for retraining, a method proven effective in literature [6, 46]. By adopting a strategic approach to sample selection for labeling, active learning can surpass random sampling [40, 52] in performance.

Algorithm 2 Active Learning for Retraining

Input: Training dataset D_{train}, Serving dataset D_{serving}, Black-box model M, Set of DDLA D_{DDLA}
Output: Enhanced black-box model M'
Algorithm:

1: $S_{DDLA} = x | x \in D_{\text{serving}}$ and $x \in D_{DDLA}$ {Samples from DDLA within the serving data}
2: $A = (x, y_{\text{expert}}) | x \in S_{DDLA}$ {Annotate samples in S_{DDLA} by experts}
3: $y_{pred} = M(S_{DDLA})$ {Predictions by the model M for S_{DDLA}}
4: $S_{\text{retrain}} = (x, y_{\text{expert}}) \in A | y_{\text{expert}} \neq y_{pred}$ {Instances where the model's prediction is incorrect}
5: **if** $S_{\text{retrain}} \neq \emptyset$ **then**
6: $D_{\text{updated}} = D_{\text{train}} \cup S_{\text{DDLA}}$ {Form the updated training set}
7: M' = Train M using D_{DDLA} {Retrain the model}
8: **return** M' {Enhanced black-box model}
9: **end if**

In our approach, the serving data is received in batches, we use a batch query to simultaneously select multiple samples, with a focus on selecting samples from DDLA regions within the serving data. We notice that this is similar to pool-based active learning where samples are selected from a pool of unlabeled data [46]. If there's no restriction on the experts' capacity to annotate samples, we query all samples from the DDLAs. However, if there's a limit and it's lower than the total number of samples in DDLA, we then rank DDLA areas by accuracy from lowest to highest and sequentially retrieve all samples from each area until we reach the maximum number of annotable samples. We denoted these samples as $S_{DDLA} = \{x | x \in D_{\text{serving}} \text{ and } x \in D_{DDLA}\}$.

Selecting samples from DDLA areas is highly informative because these regions typically highlight the model's weaknesses, and focusing on them enables more effective improvements and adaptations of the model.

These samples are then provided to domain experts for annotation, ensuring the procurement of accurate and reliable labels. This is formalized as: $A = \{(x, y_{\text{expert}}) | x \in S_{DDLA}\}$, where y_{expert} denotes the expert-annotated labels. The set A embodies the subset of instances within DDLA and their expert annotations.

Subsequently, we compare these expert annotations y_{expert} with the model's predictions y_{pred} and select the instances where the model's prediction is incorrect: $S_{\text{retrain}} = \{(x, y_{\text{expert}}) \in A | y_{\text{expert}} \neq y_{pred}\}$. Then, we check if $S_{\text{retrain}} \neq \emptyset$. If it is empty, we don't need to retrain the model. The retraining process combines the newly annotated instances S_{DDLA} with the original training dataset D_{train} to produce an updated training set: $D_{\text{updated}} = D_{\text{train}} \cup S_{\text{DDLA}}$.

This updated dataset is used to improve the model, ensuring it adapts to a broader range of data distributions. Our process of targeted sampling, annotation, and retraining leads to continuous improvement in model performance.

With these steps in mind, our approach includes:

- **DDLA-Full method,** which focuses on detecting all the cases of harmful drift, including those that are unlikely to be observed in practice (case 3 in Table 1). The steps of DDLA-Full, as shown in Figure 2, are Step 1-a, Step 1-b, Step 2, and then Step 3.

- **DDLA-Opt method,** which focuses on detecting harmful drift related to cases of drift from low accuracy distributions with low sample size to low accuracy distributions with high sample size (Case 4 in Table 1). The steps of DDLA-Opt, as shown in Figure 2, are Step 1-b, Step 2, and then Step 3. Notice that DDLA-Opt only uses the existing labeled data to identify DDLAs, and doesn't have Step 1-a to collect information from serving data, meaning it has a lower cost, as we will demonstrate in the experimental section. Additionally, we will show that its accuracy is comparable to the accuracy of DDLA-Full.

DDLA-Opt can detect the primary DDLAs and most instances of harmful data drift. DDLA-Full can identify potential DDLAs that are not visible from the existing labeled data, but these scenarios are rare. Therefore, under normal circumstances, we recommend using DDLA-Opt which has a low cost and comparable accuracy.

5 EXPERIMENTAL EVALUATION

5.1 Experimental Setup

We use Python 3.9 for all our experiments. We evaluate our approach on several datasets and under different ML models and compare its performance with baselines.

5.1.1 Datasets. We experiment on real-world datasets of diverse types and dimensions. (1) The house sales dataset [2] contains 21.6k entries with 32 attributes, detailing house sale prices. It is segmented into six two-decade periods based on construction year for drift analysis, using 1900-1919 data for training. (2) The UCI adult dataset [8] contains 48.8k entries, and 14 attributes, focused on predicting annual earnings over 50K. Split into ten groups by race and sex, training in one group. (3) CER smart meter dataset [13] comprises 4225 time-series data of half-hourly power consumption and 1000 dimensions. Used to predict dryer status, with training data from summer and serving data from winter, showcasing seasonal drift. (4) Mechanical MNIST [54], image data represented in 784 dimensions, which includes finite element simulation results on heterogeneous materials under deformation. Training from two environments, tested on two different environments to demonstrate data drift.

5.1.2 ML Models. For our experiments, we leverage the following black-box models: Logistic Regression [9], Neural Network (3-layer MLP) [38] and SVM [5].

5.1.3 Baselines. We evaluate our approach against the drift detection methods: K-S test [15] for statistical analysis, Maximum Mean Discrepancy with both Gaussian (MMD-O [22]) and deep learning kernels (MMD-D [34]), Classifier Two Sample Test (C2ST-S [35] and C2ST-L [11]), H-Divergence [57] for data entropy analysis, and the Detectron [21] learning-based hypothesis-test method.

We also compare our active learning query strategy with two commonly used active learning strategies: uncertainty sampling [47], and informative-diverse sampling [26]. Additionally, we also used a uniform random sampling method that maintains the same retraining iterations and the number of expert-marked labels as our DDLA-Opt method.

Note that in our active learning experiment, we use the ground truth labels of the dataset to simulate expert-marked labels (i.e., we assume that the expert's labels are correct).

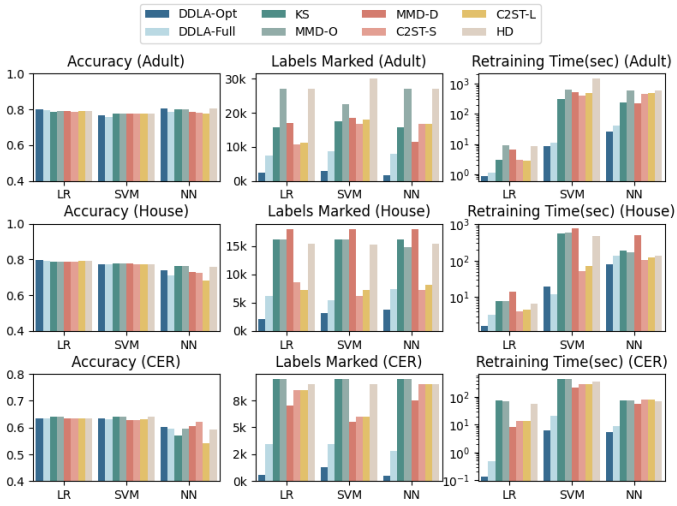


Figure 3: Histograms of Accuracy, Expert-Labeled Samples, and Retraining Time for DDLA-Full, DDLA-Opt, and other Baselines Across Adult, House, and CER Datasets.

5.1.4 Evaluation Metrics. We use these metrics for evaluation.

Effectiveness: Changes in Accuracy of ML Models. We focus on the accuracy of the updated model on the serving data following each retraining iteration. We compare the accuracy changes across different methods over one hundred batches of serving data.

Cost: Number of Expert-Labeled Samples and Retraining Time. We assess two aspects of cost: the workload associated with expert labeling in active learning and the time required for retraining. These costs are evaluated based on 100 batches of serving data.

5.2 Results

5.2.1 ***Effectiveness and efficiency of our approach compared to baselines.*** In our study, we conducted comprehensive experiments using both tabular data (Adult and House datasets) and time series data (CER dataset), simulating real-world scenarios that commonly face data drift problems. Figure 3 showcases the results, demonstrating the effectiveness and efficiency of our approach, including DDLA-Full which uniform random samples 20% of one batch of serving data in Step 1-a and DDLA-Opt in the figures.

DDLA-Full and DDLA-Opt consistently match the accuracy of compared baselines, require fewer expert labels, and boast quicker retraining times across the three datasets. This highlights our approach's superior handling of data drift, particularly in its ability to efficiently retrain models using data from low-accuracy areas.

When comparing DDLA-Full with DDLA-Opt, the expert-labeled data in DDLA-Opt is only 33.91% of that in DDLA-Full, and the training time is 77.36%. However, their accuracy is comparable with no significant difference between them.

When comparing DDLA-Opt with other methods, for the three datasets, DDLA-Opt required significantly fewer expert-labeled samples (around 13.79% of the KS, 12.28% of the HD, 12.39% of the MMD-O, 14.88% of the MMD-D, 23.48% of C2ST-S, and 22.31% of C2ST-L) and substantially reduced retraining times (only 10.55% of

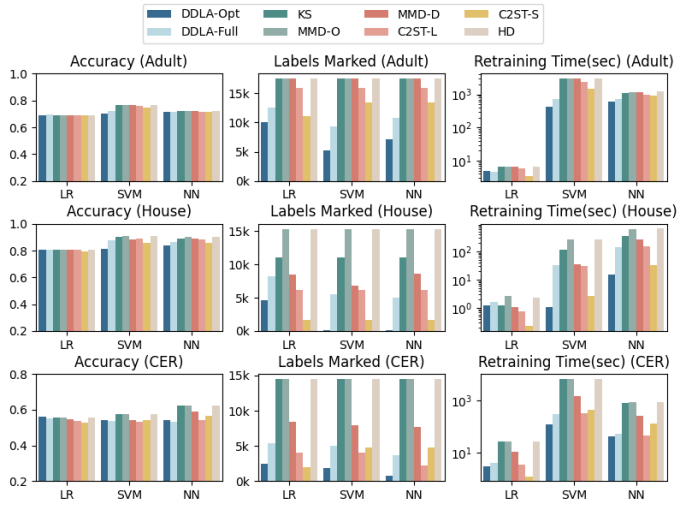


Figure 4: Stress test: Comparison of Accuracy and Cost between DDLA-Full and DDLA-Opt under the worst case where each batch of serving data has harmful data drift and gives rise to new distributions.

the time required by KS, 11.39% by HD, 9.92% by MMD-O, 6.50% by MMD-D, 20.02% by C2ST-S, and 16.93% by C2ST-L).

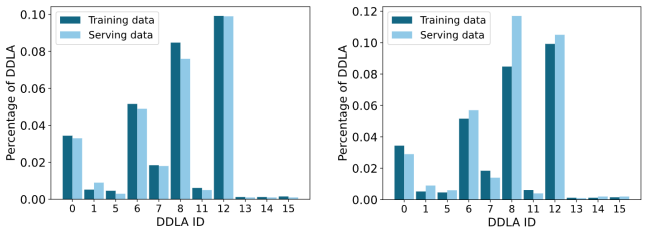
5.2.2 ***Stress testing under extreme scenarios.*** We set up a stress test where training and serving data are generated using a method that ensures almost half the distributions in the serving data are not present in the training data to present the extreme scenario where each batch of serving data has harmful data drift and gives rise to new distributions(new values). This result is the worst-case performance of our approach. We categorize all the original labeled data into DDLA and non-DDLA regions for three models—LR, SVM, and NN— ensuring that the distributions of these two data subsets are entirely non-overlapping. From the non-DDLA regions, 80% of the data is used as the training set for the black-box model, while the remaining 20% from the non-DDLA regions and 100% of the data from the DDLA constitute the serving data.

As we can see in Figure 4, DDLA-Full results in higher accuracy than DDLA-Opt in four instances, while the results are roughly equivalent in five instances. However, the cost of DDLA-Full is consistently higher than that of DDLA-Opt. The results indicate that when the case that serving data gives rise to new distributions(new values) occurs frequently, DDLA-Full indeed outperforms DDLA-Opt in improving model accuracy. Even under the least ideal conditions of serving data, DDLA-Full maintains accuracy comparable to other methods across six test scenarios. One important benefit is that, in terms of the retraining time, DDLA-Full outperforms other benchmark methods.

5.2.3 ***Identification of Black-box model's DDLAs and Determining Harmful Data Drift.*** Table 2 displays 3 key DDLAs from the Adult dataset using logistic regression, where accuracies deviated from the model's overall accuracy of 0.878. For instance, DDLA with ID 16 highlights a specific group working over 84 hours

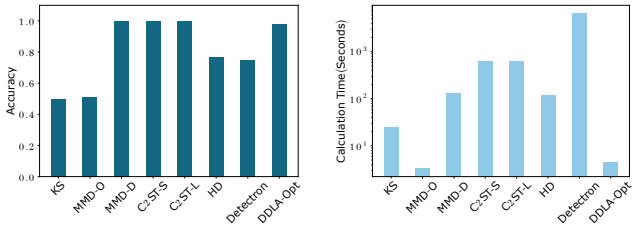
Table 2: DDLAs examples from the Adult dataset.

ID	DDLA Distribution	Acc.
11	marital = Married-civ-spouse ∧ age>25 ∧ capital_loss≤1867 ∧ capital-gain≤6431 ∧ occupation ∈ {Adm-clerical, Armed-Forces, Craft-repair, Exec-managerial}	0.513
14	marital ≠ Married-civ-spouse ∧ education_num>13 ∧ hours_week≤84 ∧ capital_loss>834	0.250
15	marital ≠ Married-civ-spouse ∧ hours_week>84	0.400



(a) DDLA ratios decrease. **(b) DDLA ratios increase.**

Figure 5: The DDLA ratios identified from the adult dataset between the training set and two serving batches.



(a) Accuracy. **(b) Computation time.**

Figure 6: Comparison of different approaches for detecting harmful drift in terms of accuracy and computation time, on the Mechanical MNIST dataset.

weekly, not 'Married-civ-spouse', pinpointing a demographic with notably lower model accuracy.

Determining Harmful Data Drift. From the DDLAs identified, we distinguish two cases. *Decreasing DDLA Ratios:* Our analysis of the Adult dataset reveals a lower DDLA ratio in the serving data compared to training data (Figure 5a). The model, tested on this serving data, showed an improved accuracy of 0.886, up from the original 0.878. This indicates that retraining is not necessary, affirming our method's effectiveness in assessing retraining needs. *Increasing DDLA Ratios:* In contrast, an increased DDLA ratio in serving data (Figure 5b) suggests the need for retraining, as evidenced by a drop in model accuracy to 0.793 from 0.878.

5.2.4 ***Effectiveness of the identified DDLAs to identify harmful data drift.*** Our experiment on the Mechanical MNIST dataset involved two training environments (e=100 and then =90). 50 batches of serving data from e = 90, showed benign drift due to previous training in this environment. Another 50 batches from e = 50, exhibited harmful drift, due to the absence of training data from this

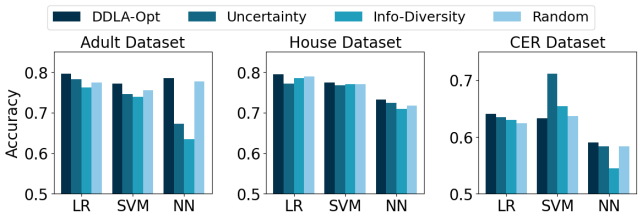


Figure 7: Comparative Accuracy of Models Using DDLA Sampling vs. Uncertainty, Informative-Diverse, Random Sampling across Datasets and Models.

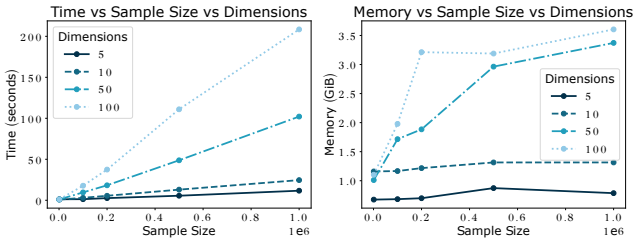


Figure 8: Scalability Analysis: Time and Memory Across Various Dimensions and Sample Sizes in CER Time-Series Data.

environment. We use different data drift detection methods to repeatedly detect the serving data of 100 batches without retraining the black box model. Figure 6a shows these results.

In 100 trials of drift detection, K-S, and MMD-O displayed an accuracy of around 0.5, failing to differentiate between benign and harmful drift. The HD method showed better performance with an accuracy of 0.77. Notably, although MMD-D, C2ST-S, and C2ST-L achieved high accuracy levels, Figure 6b reveals that their computation time was two orders of magnitude longer compared to our method. Our optimized approach, DDLA-Opt, excelled achieving an impressive accuracy of 0.98, effectively distinguishing 48 out of 50 benign drifts and thereby reducing unnecessary retraining and computational overhead. Regarding the Detectron [21] method, we observe that it exhibits significant resource consumption in our context, exceeding 6000 seconds in runtime, with an accuracy of 0.75. Considering that its runtime is another order of magnitude slower than the other methods, note that we do not consider it in our experiments.

5.2.5 ***Effectiveness of our Active Learning query strategy compared to baselines.*** In this experiment, We compare our active learning query strategy against *uncertainty* [47], and *informative-diverse* [26] baselines, and a uniform random sampling method (Random). We use Google's active learning package [1] to conduct the baselines. To evaluate the impact of active learning, we analyzed the iterations and active learning components separately. Specifically, for the baseline active learning query strategy, we used the DDLA-Opt method to detect harmful data drift and maintained the same retraining iterations as DDLA-Opt. We also kept the number of query samples for each batch of serving data consistent with our DDLA query. This setup maintained consistent retraining iterations and the same number of expert-marked labels across the other three baselines as used in our DDLA-Opt method, to allow a clear comparison of accuracy improvements for the ML model.

The results, plotted in Figure 7, show the effectiveness of our query strategy compared to the baselines. Specifically, in the nine distinct cases, the DDLA-Opt method demonstrates superior accuracy in eight cases. There is one case where the DDLA-Opt method underperforms compared to baselines.

In situations where serving data is affected by harmful data drift, the low-accuracy query strategy more effectively pinpoints and addresses the model's current weaknesses. By focusing on correcting prediction errors, this strategy allows the model to quickly adjust to new data distributions. In contrast, uncertainty sampling which prioritizes information gain, often misses the drifts in data distributions. Similarly, despite the informative and diverse strategy that aims to enhance model generalization by selecting varied and informative samples, it lacks the precision needed to tackle the specific errors that are crucial in overcoming the challenges posed by harmful data drift. Consequently, the low-accuracy query strategy proves to be more adaptable and efficient in mitigating the effects of harmful data drift.

5.2.6 ***Scalability analysis.*** We evaluated the scalability using the CER dataset, varying dimensions and sample sizes. We focused on processing time and memory usage to understand scalability. The results, shown in Figure 8, indicate a linear increase in both processing time and memory usage across various dimensions and sample sizes. This demonstrates the scalability of our method. The moderate increase in processing time can be attributed to the use of decision tree models, known for their quick training times, which significantly mitigates the impact of increased data dimensionality and size. These observations underscore the capability of our method to effectively handle large-scale datasets in the real world.

6 CONCLUSIONS

In this paper, we proposed a novel framework to mitigate the impact of potential data drift that can arise in serving data, which may cause an accuracy drop in ML models. Our approach is based on identifying DDLAs from the black-box model's predictions using decision trees. We evaluated the performance of our proposed approach with various datasets and models and compared it against drift baselines. The results demonstrate that our approach can be considerably faster than the competitors while achieving accuracy that is comparable to the best competitor.

In future work, we intend to explore the use of other parsimonious interpretable models (such as Explanation Tables [20]) to identify DDLAs. We also plan to investigate concept drift and explore the utility of DDLAs for this task. Finally, we would like to see how our approach can be used to improve model fairness in the presence of data and concept drift, when the fairness properties are satisfied on training data but violated on serving data.

ACKNOWLEDGMENTS

This work was supported by the Université Paris Cité mobility grant to support the visit of Sijie Dong to AT&T, the Chinese Scholarship Council, and the French National Agency ANR-21-CE23-0036-01.

REFERENCES

[1] 2017. Active Learning Playground. https://github.com/google/active-learning/tree/master Last accessed on July 28, 2024.

[2] 2019. House Sales in King County Data . https://www.kaggle.com/datasets/harlfoxem/housesalesprediction. Last accessed on July 28, 2024.

[3] Samuel Ackerman, Eitan Farchi, Orna Raz, Marcel Zalmanovici, and Parijat Dube. 2020. Detection of data drift and outliers affecting machine learning model performance over time. *arXiv preprint arXiv:2012.09258* (2020).

[4] Samuel Ackerman, Orna Raz, Marcel Zalmanovici, and Aviad Zlotnick. 2021. Automatically detecting data drift in machine learning classifiers. *arXiv preprint arXiv:2111.05672* (2021).

[5] S Amarappa and SV Sathyanarayana. 2014. Data classification using Support vector Machine (SVM), a simplified approach. *Int. J. Electron. Comput. Sci. Eng* 3 (2014), 435–445.

[6] Philip Bachman, Alessandro Sordoni, and Adam Trischler. 2017. Learning algorithms for active learning. In *international conference on machine learning*. PMLR, 301–310.

[7] Lucas Baier, Josua Reimold, and Niklas Kühl. 2020. Handling Concept Drift for Predictions in Business Process Mining. *2020 IEEE 22nd Conference on Business Informatics (CBI)* 1 (2020), 76–83.

[8] Barry Becker and Ronny Kohavi. 1996. Adult. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5XW20, Last accessed on July 28, 2024.

[9] Ekaba Bisong and Ekaba Bisong. 2019. Logistic regression. *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners* (2019), 243–250.

[10] Lukas Budach, Moritz Feuerpfeil, Nina Ihde, Andrea Nathansen, Nele Sina Noack, Hendrik Patzlaff, Hazar Harmouch, and Felix Naumann. 2022. The Effects of Data Quality on Machine Learning Performance.

[11] Xiuyuan Cheng and Alexander Cloninger. 2022. Classification logit two-sample testing by neural networks for differentiating near manifold densities. *IEEE Transactions on Information Theory* 68, 10 (2022), 6631–6662.

[12] Kacper P Chwialkowski, Aaditya Ramdas, Dino Sejdinovic, and Arthur Gretton. 2015. Fast two-sample testing with analytic representations of probability measures. *Advances in Neural Information Processing Systems* 28 (2015).

[13] Commission for Energy Regulation (CER). 2012. CER Smart Metering Project - Electricity Customer Behaviour Trial, 2009-2010 [dataset]. Irish Social Science Data Archive. https://www.ucd.ie/issda/data/commissionforenergyregulationcer/ SN: 0012-00, Last accessed on July 28, 2024.

[14] Mengnan Du, Ninghao Liu, and Xia Hu. 2019. Techniques for interpretable machine learning. *Commun. ACM* 63, 1 (2019), 68–77.

[15] Giovanni Fasano and Alberto Franceschini. 1987. A multidimensional version of the Kolmogorov–Smirnov test. *Monthly Notices of the Royal Astronomical Society* 225, 1 (1987), 155–170.

[16] Daniele Foroni, Matteo Lissandrini, and Yannis Velegrakis. 2021. Estimating the extent of the effects of Data Quality through Observations. *2021 IEEE 37th International Conference on Data Engineering (ICDE)* (2021), 1913–1918.

[17] Nicholas Frosst and Geoffrey Hinton. 2017. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784* (2017).

[18] Joao Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues. 2004. Learning with drift detection. In *Advances in Artificial Intelligence–SBIA 2004: 17th Brazilian Symposium on Artificial Intelligence, Sao Luis, Maranhao, Brazil, September 29-Ocotber 1, 2004. Proceedings 17*. Springer, 286–295.

[19] João Gama, Indre Zliobaite, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A survey on concept drift adaptation. *ACM Comput. Surv.* 46, 4 (2014), 44:1–44:37.

[20] Kareem El Gebaly, Parag Agrawal, Lukasz Golab, Flip Korn, and Divesh Srivastava. 2014. Interpretable and Informative Explanations of Outcomes. *PVLDB.* 8, 1 (2014), 61–72.

[21] Tom Ginsberg, Zhongyuan Liang, and Rahul G. Krishnan. 2023. A Learning Based Hypothesis Test for Harmful Covariate Shift. arXiv:2212.02742 [cs.LG]

[22] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A kernel two-sample test. *The Journal of Machine Learning Research* 13, 1 (2012), 723–773.

[23] Arthur Gretton, Alex Smola, Jiayuan Huang, Marcel Schmittfull, Karsten Borgwardt, Bernhard Schölkopf, et al. 2009. Covariate shift by kernel mean matching. *Dataset shift in machine learning* 3, 4 (2009), 5.

[24] Nitin Gupta, Hima Patel, Shazia Afzal, Naveen Panwar, Ruhi Sharma Mittal, Shanmukha Guttula, Abhinav Jain, Lokesh Nagalapatti, Sameep Mehta, Sandeep Hans, et al. 2021. Data Quality Toolkit: Automatic assessment of data quality and remediation for machine learning datasets. *arXiv preprint arXiv:2108.05935* (2021).

[25] Peter Hase, Chaofan Chen, Oscar Li, and Cynthia Rudin. 2019. Interpretable image recognition with hierarchical prototypes. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, Vol. 7. 32–40.

[26] Sheng-Jun Huang, Rong Jin, and Zhi-Hua Zhou. 2010. Active learning by querying informative and representative examples. *Advances in neural information processing systems* 23 (2010).

[27] Shichao Jia, Peiwen Lin, Zeyu Li, Jiawan Zhang, and Shixia Liu. 2020. Visualizing surrogate decision trees of convolutional neural networks. *Journal of Visualization* 23 (2020), 141–156.

[28] Minsuk Kahng, Dezhi Fang, and Duen Horng Chau. 2016. Visual exploration of machine learning results using data cube analysis. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*. 1–6.

[29] Bojan Karlaš, Peng Li, Renzhi Wu, Nezihe Merve Gürel, Xu Chu, Wentao Wu, and Ce Zhang. 2020. Nearest neighbor classifiers over incomplete information: From certain answers to certain predictions. *arXiv preprint arXiv:2005.05117* (2020).

[30] Sanjay Krishnan, Jiannan Wang, Eugene Wu, Michael J Franklin, and Ken Goldberg. 2016. Activeclean: Interactive data cleaning for statistical modeling. *PVLDB* 9, 12 (2016), 948–959.

[31] Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Jure Leskovec. 2017. Interpretable & explorable approximations of black box models. *KDD'17, Workshop on Fairness, Accountability, and Transparency in Machine Learning* (2017).

[32] Peng Li, Xi Rao, Jennifer Blase, Yue Zhang, Xu Chu, and Ce Zhang. 2019. CleanML: A Benchmark for Joint Data Cleaning and Machine Learning [Experiments and Analysis]. *CoRR* abs/1904.09483 (2019).

[33] Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. 2018. Detecting and correcting for label shift with black box predictors. In *International conference on machine learning*. PMLR, 3122–3130.

[34] Feng Liu, Wenkai Xu, Jie Lu, Guangquan Zhang, Arthur Gretton, and Danica J Sutherland. 2020. Learning deep kernels for non-parametric two-sample tests. In *International conference on machine learning*. PMLR, 6316–6326.

[35] David Lopez-Paz and Maxime Oquab. 2017. Revisiting classifier two-sample tests. *ICLR* (2017).

[36] Ankur Mallick, Kevin Hsieh, Behnaz Arzani, and Gauri Joshi. 2022. Matchmaker: Data Drift Mitigation in Machine Learning for Large-Scale Systems. *Proceedings of Machine Learning and Systems* 4 (2022), 77–94.

[37] Katsiaryna Mirylenka, George Giannakopoulos, Le Minh Do, and Themis Palpanas. 2017. On classifier behavior in the presence of mislabeling noise. *Data Min. Knowl. Discov.* 31, 3 (2017), 661–701.

[38] Bo Pang, Erik Nijkamp, and Ying Nian Wu. 2020. Deep learning with tensorflow: A review. *Journal of Educational and Behavioral Statistics* 45, 2 (2020), 227–248.

[39] Aleksandr Podkopaev and Aaditya Ramdas. 2022. Tracking the risk of a deployed model and detecting harmful distribution shifts. arXiv:2110.06177 [stat.ML]

[40] Maria E Ramirez-Loaiza, Manali Sharma, Geet Kumar, and Mustafa Bilgic. 2017. Active learning: an empirical study of common baselines. *Data mining and knowledge discovery* 31 (2017), 287–313.

[41] Jie Ren, Stanislav Fort, Jeremiah Liu, Abhijit Guha Roy, Shreyas Padhy, and Balaji Lakshminarayanan. 2021. A simple fix to mahalanobis distance for improving near-ood detection. *arXiv preprint arXiv:2106.09022* (2021).

[42] Cedric Renggli, Luka Rimanic, Nezihe Merve Gürel, Bojan Karlaš, Wentao Wu, and Ce Zhang. 2021. A Data Quality-Driven View of MLOps. arXiv:2102.07750 [cs.LG]

[43] Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. 2022. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistic Surveys* 16 (2022), 1–85.

[44] Sebastian Schelter, Dustin Lange, Philipp Schmidt, Meltem Celikel, Felix Biessmann, and Andreas Grafberger. 2018. Automating large-scale data quality verification. *PVLDB* 11, 12 (2018), 1781–1794.

[45] Sebastian Schelter, Tammo Rukat, and Felix Bießmann. 2020. Learning to validate the predictions of black box classifiers on unseen data. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 1289–1299.

[46] Burr Settles. 2009. *Active Learning Literature Survey.* Computer Sciences Technical Report 1648. University of Wisconsin–Madison. http://axon.cs.byu.edu/~martinez/classes/778/Papers/settles.activelearning.pdf

[47] Manali Sharma and Mustafa Bilgic. 2017. Evidence-based uncertainty sampling for active learning. *Data Mining and Knowledge Discovery* 31 (2017), 164–202.

[48] Alaa Tharwat and Wolfram Schenck. 2023. A Survey on Active Learning: State-of-the-Art, Practical Challenges and Research Directions. *Mathematics* 11, 4 (2023), 820.

[49] Heng Wang and Zubin Abraham. 2015. Concept drift detection for streaming data. In *2015 international joint conference on neural networks (IJCNN)*. IEEE, 1–9.

[50] Geoffrey I Webb, Loong Kuan Lee, Bart Goethals, and François Petitjean. 2018. Analyzing concept drift and shift from sample data. *Data Mining and Knowledge Discovery* 32 (2018), 1179–1199.

[51] G. Widmer and M. Kubat. 1996. Learning in the Presence of Concept Drift and Hidden Contexts. *Machine Learning* 23, 1 (1996), 69–101.

[52] Hoyoung Woo and Cheong Hee Park. 2013. An efficient active learning method based on random sampling and backward deletion. In *Intelligent Science and Intelligent Data Engineering: Third Sino-foreign-interchange Workshop, IScIDE 2012, Nanjing, China, October 15-17, 2012. Revised Selected Papers 3*. Springer, 683–691.

[53] Doris Xin, Hui Miao, Aditya G. Parameswaran, and Neoklis Polyzotis. 2021. Production Machine Learning Pipelines: Empirical Analysis and Optimization

Opportunities. *Proceedings of the 2021 International Conference on Management of Data* (2021).

[54] Lingxiao Yuan, S. Park, Harold, and Emma Lejeune. 2022. Mechanical MNIST – Distribution Shift. https://open.bu.edu/handle/2144/44485 Last accessed on July 28, 2024.

[55] Jan Zenisek, Florian Holzinger, and Michael Affenzeller. 2019. Machine learning based concept drift detection for predictive maintenance. *Computers & Industrial Engineering* 137 (2019), 106031.

[56] Quanshi Zhang, Yu Yang, Haotian Ma, and Ying Nian Wu. 2019. Interpreting cnns via decision trees. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 6261–6270.

[57] Shengjia Zhao, Abhishek Sinha, Yutong He, Aidan Perreault, Jiaming Song, and Stefano Ermon. 2021. Comparing distributions by measuring differences that affect decision making. In *International Conference on Learning Representations*.