CRC Credit Research Centre

CRC Working Papers

A Novel Dynamic Ensemble Learning (DEL) Framework to Combat The Dataset Shift: The Case of Loss Given Default

By Junfeng Zhang , Galina Andreeva, Yizhe Dong

March 2025

Keywords: Dataset shift, Ensemble learning, Dynamic ensemble

THE UNIVERSITY of EDINBURGH

Contact
Credit Research Centre
University of Edinburgh Business School
29 Buccleuch Place
Edinburgh, EH8 9JS
credit.research.centre@ed.ac.uk

Title

Junfeng Zhang*, Galina Andreeva, Yizhe Dong

*Corresponding Author. Email address: credit.research.centre@ed.ac.uk

Abstract

Dataset shift presents a fundamental challenge in financial prediction, where machine learning models trained on historical data often fail to generalize, due to evolving economic conditions and structural changes in data distributions. Addressing this issue requires not only detecting and characterizing dataset shift but also developing adaptive modelling strategies that maintain predictive accuracy in dynamic environments. In this study, we propose a novel Dynamic Ensemble Learning (DEL) framework that enhances the robustness of the model under distributional shifts. First, we develop a feature-level dataset shift detection methodology to quantify changes in input distributions and assess their impact on model generalization. We further introduce a continuous monitoring system based on a sliding-window approach, al lowing real-time tracking of evolving data distributions. To mitigate dataset shift, we propose four dynamic ensemble learning strategies that adaptively integrate base learners either hori zontally (by sample distances) or vertically (by feature statistics). Using Loss Given Default (LGD) modelling as a case study, we demonstrate that non-parametric models exhibit greater resilience to dataset shift, and our feature-based dynamic ensemble strategy significantly out performs traditional static ensembles and single models. Our findings provide actionable in sights for financial institutions to enhance predictive accuracy, reduce model risk, and improve long-term decision-making in dynamic environments

1 Introduction

Machine learning models rely on the fundamental assumption that patterns learned from historical data will generalize to unseen data. However, in reality, this assumption is often violated due to dataset shift, systematic changes in data distributions over time (Qi et al. 2022). These shifts degrade model performance, increasing prediction errors and model risk (Simester et al. 2020). This problem is widespread in all applications of predictive modeling, but is particularly serious in finance, where inaccurate forecasts translate into direct losses and potential regulatory non-compliance. Despite the growing use of machine learning in financial modeling, existing approaches primarily focus on improving predictive accuracy within static environments, offering limited adaptability to evolving data distributions.

To better deal with dataset shift problem, we propose a comprehensive framework for detecting, monitoring, and mitigating dataset shift in financial predictions. First, we develop a feature-level dataset shift detection methodology to systematically quantify distributional changes between training and testing data. Using statistical tests such as the Kolmogorov-Smirnov test and Maximum Mean Discrepancy, we analyze how different types of features evolve over time. Second, we introduce a sliding-window monitoring system, which continuously tracks evolving distributions, providing real-time insights for financial institutions to adjust models dynamically. Unlike traditional static evaluations, our monitoring approach ensures continuous visibility into dataset drift, allowing for proactive model adjustments before performance deteriorates. To demonstrate the dataset shift problem and validate our proposed framework, we use Loss Given Default (LGD) modeling as a case study. LGD data is known to be difficult to observe, and existing datasets with sizable samples typically span long time periods, increasing the likelihood of dataset shift. As a result, models trained on historical LGD data often struggle to generalize to new time periods. Existing studies have reported that the Out-Of-Time (OOT) performance of LGD models is significantly worse than their Out-Of-Sample (OOS) performance (Kalotay & Altman 2017, Nazemi et al. 2022).

We conduct a comprehensive investigation of dataset shift and the stability of different machine learning models when exposed to such shifts. We find that in LGD modeling, model performance deteriorates significantly on unseen data, with parametric models such as linear regression and deep neural networks experiencing severe out-of-range prediction issues. Specifically, parametric models tend to overestimate LGD, while non-parametric models tend to underestimate it. We attribute this phenomenon to differences in how models handle extreme values during training. Furthermore, to better tackle dataset shift problem, it is important to understand the type of the shift. Following the methodology of Rabanser et al. (2019), we measure distributional differences across features in training and testing datasets, revealing that covariate

shift plays a more dominant role than concept shift in LGD modeling.

To mitigate dataset shift, we propose a Dynamic Ensemble Learning (DEL) framework, designed to enhance model robustness by adapting to changing data patterns. Our framework consists of four dynamic ensemble strategies, including one meta-learning-based approach and three rule-based approaches. The rule-based approaches are further categorized into two integration methods: (i) horizontal integration, which adapts to dataset shift by using sample distances in the vector space, and (ii) vertical integration, which adjusts model contributions based on feature statistics. Specifically, the META-DESR (Meta-Learning-Based Dynamic Ensemble for Regression) utilizes a meta-learner to estimate model competence dynamically, ensuring optimal model selection under evolving conditions. By dynamically adjusting model weights based on evolving data characteristics, our approach significantly outperforms both individual models and static ensemble methods. Our empirical results demonstrate that non-parametric models are more resilient to dataset shift, and that feature-based dynamic ensemble strategies are particularly effective when spatial structures are unstable, but certain feature relationships remain invariant over time.

Beyond LGD modeling, our study extends the application of dynamic ensemble learning by proposing multiple effective strategies compatible with regression tasks, which can be generalized to other financial and predictive modeling applications facing dataset shift. A common approach to addressing dataset shift is frequent model retraining, but this leads to higher maintenance costs. Our framework offers a more efficient alternative, allowing models to dynamically adapt without requiring constant manual intervention. This work contributes to the discussion of model uncertainty as well. By systematically detecting and monitoring dataset shift, we quantify the instability in commonly used machine learning models in the context of LGD modeling, addressing both aleatoric uncertainty from macroeconomic fluctuations and epistemic uncertainty due to model limitations.

This research further aligns with the broader movement in finance toward distributional robustness and adaptive modeling (Anderson & Cheng 2022, Blanchet et al. 2022, Dubiel-Teleszynski et al. 2024, Si et al. 2023). Our DEL framework innovatively blends adaptive model selection with robust principles, enabling real-time recalibration of forecasts in the face of shifting market conditions. This adaptability not only refines risk estimation, which is crucial for managing credit risk and regulatory compliance, but also offers a scalable solution for deployment in volatile environments. The framework's flexible architecture holds promise for extension to other financial modeling tasks, including asset pricing and portfolio optimization, thereby providing a practical tool for enhancing decision-making across the financial industry.

This paper is structured as follows. Section 2 provides a literature review on dataset shift detection, monitoring techniques, and ensemble learning methods. Section 3 presents our methodology, detailing

the dataset shift detection framework, continuous monitoring approach, and the proposed DEL framework. Section 4 describes the experimental setup and results, outlining the dataset, model evaluation metrics, and comparative performance analysis of different ensemble learning strategies under dataset shift. Section 5 concludes the study, summarizing key findings, discussing implications for financial risk modeling, and proposing future research roadmap.

2 Literature Review

2.1 Dataset Shift

Dataset shift is a common problem that people would face when doing machine learning modeling, where the distribution of the data changes between the training set and the test set. According to Moreno-Torres et al. (2012), the dataset shift can be defined as

$$P_{training}(y, \mathbf{x}) \neq P_{testing}(y, \mathbf{x})$$

There are different types of dataset shift. First of all, according to the cause of the shift, we can divide it into $X \rightarrow Y$ problem and $Y \rightarrow X$ problem. In general, $Y \rightarrow X$ problem is common in medical/healthcare related machine learning projects, since the diseases that we are interested in determining the specific symptoms on the patients. While in the credit risk domain, it is the macroeconomic environment or the companies' financial performance and business strategies that ultimately impacts their probability of default or credit ratings. Therefore, $X \rightarrow Y$ problem applies here, which is also the main focus of this paper.

Within $X \rightarrow Y$ dataset shift problem, it can be further categorised into three cases. First, covariate shift, which is most commonly observed situation, indicates the distribution shift of the input variables X, while the conditional probability of Y given X remains the same, i.e.,

$$P_{training}(y|\mathbf{x}) = P_{testing}(y|\mathbf{x}),\ P_{training}(\mathbf{x}) \neq P_{testing}(\mathbf{x})$$

Second, concept shift refers to a worst scenario that although the distribution of covariate X remains the same but the conditional probability of Y given X has changes, i.e.,

$$P_{training}(y|\mathbf{x}) \neq P_{testing}(y|\mathbf{x}),\ P_{training}(\mathbf{x}) = P_{testing}(\mathbf{x})$$

Third, both conditional distribution of Y given X and the distribution of the covariate X have changed, i.e.,

$$P_{training}(y|\mathbf{x}) \neq P_{testing}(y|\mathbf{x}),\ P_{training}(\mathbf{x}) \neq P_{testing}(\mathbf{x})$$

Following the exploration of dataset shift types, the researchers and industry practitioners propose plenty of approaches for detection and adaptation (Quinonero-Candela et al. 2008). The detection of dataset shifts typically involves methods like statistical tests and divergence measures. If necessary, for some high-dimensional data, dimensionality reduction methods should be applied before the detection. Rabanser et al. (2019) provide an empirical study of various detection methods, emphasising the importance of identifying shifts in data distributions using statistical tests and discrepancy measures, including Kolmogorov–Smirnov statistics, maximum mean discrepancy, and black box shift estimation proposed by Lipton et al. (2018). Rabanser et al. (2019) also compared the impact of different dimensional reduction techniques on the dataset shift detection tests, providing a simple and effective detection framework.

So far, multiple directions are pointed out by researchers to handle the dataset shift problem. The regularisation techniques are indeed one of them. Regularisation methods, such as L1 and L2 regularisation, dropout in neural networks, and early stopping during training, prevent overfitting to the training data, making the model more generalisable to the new data (Tian & Zhang 2022). However, regularisation techniques sometimes are applied merely for forcing model not to learn the noise of the training data. Also too strong regularisation can result in a over-simplified model which is still not competent in predicting over new data. Also, some data preprocessing steps and feature selection methods can help, by identifying and utilising features that are invariant across different distributions can reduce the impact of dataset shift, which is usually combined with the feature-level distributional shift detection (Bolón-Canedo et al. 2016). Alternatively, in particular to the covariate shift with independent variables for both testing set and training set available, Shimodaira (2000) uses importance weighting to address this issue by adjusting the weights of the training samples to compensate for the distributional discrepancy, thereby improving the model's generalisation ability, which emphasises the theoretical foundation of importance weighting in addressing covariate shift by weighting the likelihood function. Afterwards, many extensions and improvements were proposed on this direction, making importance weighting more accurate and compatible with deep learning methods (Sugiyama et al. 2007, Sugiyama & Kawanabe 2012, Fang et al. 2020, Li et al. 2020). Another proven promising method for dataset shift mitigation is ensemble learning, which is versatile and can be combined with other types of dataset shift mitigation approaches flexibly. In the next subsection, we will review the ensemble learning and discuss how it can effectively combat distributional shift.

2.2 Ensemble Learning and Dynamic Ensemble Selection

Ensemble learning has emerged as a pivotal approach in tackling the challenge of dataset shift in machine learning, which involves combining multiple models to diversify the decision process and thus improve the robustness and accuracy of predictions. The ensemble learning is very flexible. First, there are multiple strategies to construct an ensemble. If we focus on the models in the ensemble, the strategies can be divided into homogeneous ensembles and heterogeneous ensembles. Homogeneous ensembles use the same base learning algorithm to produce multiple models with variations, e.g., Bagging (Bootstrap Aggregating) and boosting (Breiman 2001, Chen & Guestrin 2016, Ke et al. 2017). Bagging involves training multiple models in parallel on different subsets of the data (created with replacement) and then averaging their predictions to improve the overall model's accuracy and stability. Boosting, on the other hand, trains models sequentially, with each model focusing on the errors made by the previous ones, adjusting the weights of incorrectly predicted instances to enhance the model's ability to handle difficult cases, thereby incrementally improving performance. Heterogeneous ensembles combine different types of models to capture a broader range of data patterns by exploiting the different fitting inclination and bias-variance properties of the diverse models, often using a combination technique known as stacking or stacked generalisation. Also, depending on which aspects of the data we look at, there are feature-based ensembles and data-based ensembles. Feature based ensembles partition the feature space, training different models on distinct subsets of features. This method is particularly beneficial in handling high-dimensional data, as it allows each model to specialise in different features, reducing the complexity. While the data-based ensembles trains models on different partitions of the dataset but with all features, which can effectively reduces variance and overfitting (Breiman 2001).

Moreover, there are various ensemble integration techniques to generate the final output from the base learners, which can be broadly divided into static and dynamic methods. Static methods involve fixed rules for combining base learner predictions, regardless of the test instance or specific characteristics of the learners. Typical static integration methods includes simple averaging/voting or weighted averaging/voting. On the other hand, dynamic methods adjust the integration strategy based on the characteristics of each test instance or the performance of base learners. This paradigm acknowledges that different learners in an ensemble may perform better on different subsets of the feature space (Britto Jr et al. 2014). It can either select the most competent model or use multiple base learners with different weights applied according to their competencies, which is termed as dynamic ensemble. The concept of dynamic ensemble was initially explored as a way to improve the modelling performance by leveraging the strengths of individual models in an ensemble. Dynamic ensemble has proven particularly useful in scenarios with non-stationary data dis-

tributions or where different parts of the input space have different characteristics (Britto Jr et al. 2014). The dynamic selection or weighting process of the ensemble can be powered by various approaches, including rule-based methods (Guo et al. 2021, Yu et al. 2023), meta-learner (Cruz et al. 2015), Bayesian optimisation (Du et al. 2022), or deep reinforcement learning (Saadallah & Morik 2021). Different methods have their own pros and cons, e.g., rule-based methods are generally simpler with lower computation overhead and more intepretability, while the meta learner or deep RL method is more powerful in finding the hidden patterns embedded in the meta features.

The core idea of dynamic ensemble is to evaluate the competence of each learners for a given test instance and select only those deemed most capable or assign them a higher weight while reduce the weight of incapable learners(Cruz et al. 2015). Significant methodological contributions have been made in dynamic ensemble learning, particularly in competence estimation methods. One common approach involves estimating the competence of the learners based on their performance in a region of the feature space surrounding the test instance, often determined by k-nearest neighbours (Cruz et al. 2018). META-DES, proposed by (Cruz et al. 2015) and referring to a dynamic ensemble selection (DES) framework using meta-learning, innovatively utilises a meta-learning algorithm to evaluate the competence of individual classifiers, analysing various features extracted from the region surrounding each query sample and employing multiple criteria for classifier selection and integrating a two-level validation strategy to enhance prediction reliability and accuracy. Similar nearest neighbour based dynamic ensemble techniques also include K-nearest oracles eliminate, DES-KNN, DES-Clustering, etc. (Cruz et al. 2020). Dynamic ensemble techniques are also widely applied to financial prediction tasks especially with time series prediction and credit scoring. Hou et al. (2020) integrate the oversampling technique and the weighting mechanism for multi-class into seven different DES algorithms to improve the performance of DES on imbalanced datasets, achieving lower Type I error rate than popular boosting methods on real-world P2P loan data. Yu et al. (2023) even propose a weight random selection with confidence (WRSC) strategy, a dynamic ensemble of reinforcement learning models, based on the model confidence to optimise the stock portfolio return, revealing superior performance on the datasets of three major stock markets.

However, in general, dynamic ensemble is predominantly used in classification tasks rather than regression, with the only exception of time series data. The competence of models in dynamic ensemble is typically evaluated within specific regions of the data space. But for regression, this local competence assessment becomes more challenging due to the complexity of continuous outcome variables, as indicated by Sagi & Rokach (2018). And the regular panel data, different from time series data which are auto-correlated and be able to model from the lagged terms, can raise additional problems in rule or feature design for the

dynamic ensemble strategy.

3 Methodology

3.1 Dataset Shift Detection

Firstly, here we present the dataset shift detection framework, following the detection framework design of Rabanser et al. (2019). We split the dataset according to chronological order into training set and testing set with the ratio 7:3. The training set covers the time period from 1987 to 2009, while the testing set covers the time period from 2010 to 2020. We use Kolmogorov-Smirnov (KS) test as the baseline technique to detect the dataset shift. The two-sample Kolmogorov-Smirnov test is designed to test the null hypothesis that two independent samples are drawn from the same continuous distribution. Unlike many other statistical tests, the KS test does not assume a specific distribution shape (e.g., normal distribution), making it a non-parametric and distribution-free method. Mathematically, if we consider two cumulative distribution functions (CDFs), $F(x)$ for the first sample and $G(x)$ for the second sample, the KS statistics D can be defined as the maximum absolute difference between these two CDFs:

$$D = \max_x |F(x) - G(x)|$$

KS test is a univariate testing, therefore, with a dataset with K features, we should conduct KS test for each of the features. To maintain the best interpretability of the result, we do not use any dimensional reduction techniques and all the tests are based on the original features. Also, in order to generate an overall metric to examine the whole data instead of only certain columns, we use the Bonferroni correction. The Bonferroni correction is a widely used method in statistical analysis to address the problem of multiple comparisons (Armstrong 2014). When conducting several statistical tests simultaneously, the probability of a Type I error (falsely rejecting at least one true null hypothesis) increases. The Bonferroni correction is a conservative approach to control the family-wise error rate, which is the probability of making one or more Type I errors across all the tests. The Bonferroni correction adjusts the significance level based on the number of tests being performed. If m independent tests are conducted and we desire an overall significance level α, the Bonferroni correction suggests testing each individual hypothesis at a significance level of α/m.

Alternatively, to better evaluate the level of distributional shift, apart from KS test, we also introduce Maximum Mean Discrepancy (MMD) multivariate two-sample test. Originating from the domain of kernel methods, MMD offers a way to quantify differences between distributions in a feature space defined by

a kernel function (Gretton et al. 2012). The core idea of MMD is to compare the mean embeddings of two distributions in a Reproducing Kernel Hilbert Space (RKHS). For two distributions P and Q, MMD is defined as the distance between the means of these distributions in the RKHS:

$$\text{MMD}(P, Q) = ||E_{x\sim P}[k(x, \cdot)] - E_{y\sim Q}[k(y, \cdot)]||^2_{\text{RKHS}}$$

where $k(x, y)$ is a kernel function which implicitly maps the data to a high-dimensional space, facilitating the detection of complex distributional differences. Particularly, among all independent variables, TYPE and RECOVERY_INDUSTRY are categorical variables. In order to make these two features suitable for KS and MMD test, we first use one-hot encoding to create a sparse matrix for each feature and embed the sparse matrix into a dense continuous value via an autoencoder. An autoencoder is a type of neural network used to learn efficient representations of data, typically for dimensionality reduction. In the context of sparse matrices, it transforms high-dimensional, sparse input into a lower-dimensional, dense representation, achieved through a process where the encoder compresses the input and the decoder attempts to reconstruct it, resulting in a continuous, compact representation of the original data (Tschannen et al. 2018).

3.2 Ensemble Learning Techniques

In order to conduct a comprehensive comparison study and reveal how well other static and dynamic ensemble learning techniques perform on the LGD dataset, we will compare seven different types of ensemble learning techniques, covering three static methods and four dynamic methods. For static ensembles, we choose three different strategies:

1. Heterogeneous base learner ensemble: We assume different models can capture different relationships in the data, including linear, non-linear and rule-based relationships, to make the ensemble perform better than any single base model. This ensemble combines all different base learners examined in the study with a simple average as the output.

2. Static bagging ensemble: With predetermined hyperparameters the number of the base regressor and the resampling ratio, the algorithm first extracts subsets from the original dataset with replacement (but in each subset the resampling is without replacement to avoid overfitting) and train multiple regressors to make sure different learners can capture diverse relationships embedded in different parts of the data.

3. Feature-based ensemble: due to the nature of the dataset shift, some of the features may be consistent over time and provide good predictive power while the others not. In this ensemble strategy, for each base model, only a part of the features are selected to fit with a predetermined hyperparameter 'feature_ratio'. The purpose of this ensemble strategy is to let each base learner to find and fit some simple but effective relationships embedded in the data.

The four dynamic ensemble learning techniques include:

1. K-Means distance-based bagging ensemble.
2. Dynamic distance-based bagging ensemble.
3. META-DESR: A meta-learning based dynamic weighting ensemble for regression.
4. Feature-based dynamic ensemble.

The core idea and the pseudo-code of the four dynamic ensemble strategies will be illustrated below. First, this K-Means cluster distance based dynamic ensemble selection is adopted and extended from the study of Guo et al. (2021) by migrating the idea to regression task. We adjust the rule of calculating distance and the output integration to make it suitable for our regression task. Before fitting the base learners, a K-Means clustering algorithm is conducted on the training set to find k data clusters. And the same number of base learners are initialized and fitted accordingly. In the integration part, the algorithm calculates the euclidean distance d_i between the independent vectors of the current instance and the data clusters. In our implementation we choose to use the minimum distance instead of average distance. Then the weights are assigned accordingly in the form of $w_i = \frac{1/d_i}{\sum_{i=1}^{k} 1/d_i}$.

Algorithm 1 K-Means Clustering Distance-Based Dynamic Ensemble Strategy

procedure TRAINKMEANSENSEMBLE(D_{tr}, K)
  Input: Training data D_{tr}, number of clusters K
  Output: Ensemble of trained learners $R = \{R_1, R_2, \ldots, R_K\}$
  $\{C_1, C_2, \ldots, C_K\} \leftarrow$ KMEANSCLUSTERING(D_{tr}, K)
  for $i = 1$ to K **do**
    Initialize base learner R_i
    TRAIN(R_i, C_i)
  end for
  return $R = \{R_1, \ldots, R_K\}$ ▷ Trained ensemble
end procedure

procedure PREDICTKMEANSENSEMBLE($D_{te}, R, \{C_1, \ldots, C_K\}$)
  Input: Testing data D_{te}, trained ensemble R, and clusters $\{C_1, \ldots, C_K\}$
  for $i = 1$ to $|D_{te}|$ **do** ▷ Iterate over test instances
    for $k = 1$ to K **do**
      $d_{ik} \leftarrow$ DISTANCE(D_{te_i}, C_k)
      $\text{prediction}_k \leftarrow$ PREDICT(R_k, D_{te_i})
    end for
    COMPUTEWEIGHTS($\{d_{i1}, \ldots, d_{iK}\}$) ▷ Based on distances
    $\hat{y}_i \leftarrow$ INTEGRATEPREDICTIONS($\{\text{prediction}_1, \ldots, \text{prediction}_K\}$)
  end for
  return $\{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{|D_{te}|}\}$ ▷ Final predictions
end procedure

Second, we develop a new distance-based dynamic ensemble selection algorithm. Different from the previous one, this algorithm does not calculate the K-Means clusters, but simply use Bagging method to draw subsets from the training dataset randomly, aiming to introduce some randomness in the algorithm and make the base learners even more diverse. The only hyperparameters we need to care about in this algorithm are the number of base learners, which is also the times algorithm draw subsets from the training data, and the percentage of each drawing. For the dynamic integration, this algorithm use the similar mechanism as

the previous K-Means distance based dynamic ensemble does.

Algorithm 2 Distance-Based Dynamic Ensemble Selection

procedure TRAINDBDES(D_{tr}, N, α)
  Input: Training data D_{tr}, number of learners N, bagging ratio α
  Output: An ensemble of trained learners $R = \{R_1, \ldots, R_N\}$, and a list of subsets $L = \{D_1, \ldots, D_N\}$
  $L \leftarrow [\,]$ ▷ Initialize an empty list for data subsets
  for $n = 1$ to N **do**
    $D_n \leftarrow$ SAMPLEWITHBAGGING(D_{tr}, α) ▷ Randomly sample subset
    Initialize base learner R_n
    TRAIN(R_n, D_n)
    append D_n to L
  end for
  return (R, L)
end procedure

procedure PREDICTDBDES(D_{te}, R, L)
  Input: Testing data D_{te}, ensemble R, list of subsets L
  Output: Predictions $\{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{|D_{te}|}\}$
  for $i = 1$ to $|D_{te}|$ **do**
    $x_i \leftarrow D_{te}[i]$ ▷ Current test instance
    for $n = 1$ to N **do**
      $d_{in} \leftarrow$ COMPUTEMINDISTANCE($x_i, L[n]$)
      prediction$_n \leftarrow$ PREDICT(R_n, x_i)
    end for
    $\hat{y}_i \leftarrow$ COMBINEPREDICTIONS($\{\text{prediction}_1, \ldots, \text{prediction}_N\}, \{d_{i1}, \ldots, d_{iN}\}$)
  end for
  return $\{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{|D_{te}|}\}$
end procedure

Thirdly, still on the direction of sub-setting the dataset horizontally, we propose a meta-learning based dynamic ensemble selection method for regression tasks, META-DESR, which is adopted and modified from Cruz et al. (2015). The pseudo-code of the algorithm is shown in 3. Initially, given the number of learners we want to build in the ensemble and the bagging ratio, the algorithm randomly gets subsets from the training data and use the extracted subsets to train base learners accordingly. This step ensures the diversity of the base learners, allowing different base learners to draw different embedded data relationships. Then the algorithm runs the meta-learning phase. In this stage, a meta learner is initialised, followed by the construction of meta features. In META-DESR, we exploit three groups of meta features.

1. The first group of features is prediction statistics. For each instance, the base learners make predictions, and the algorithm compute the minimum, maximum, mean and standard deviation of the current predictions. This group of features provide the consistency information of base learners on current instance. The base learners' prediction is also included.

2. The second group of features is the base learners' performance on k-nearest neighbours. Hyperparameter k is predetermined and can be tuned with validation set. The algorithm will find the k nearest neighbours of the current instance from the training set and calculate the predictive errors on the local competence samples.

3. The third group of features is the base learner's performance on samples with the similar output profile. The base learners' output on training set are saved in a matrix. Then for each output profile, the algorithm finds m samples with the most similar output profiles based on the Euclidean distance and calculate the predictive errors of the current base learner on these selected samples.

In the proposed algorithm, the meta learner in the algorithm is a regressor. In the classification setting, the meta learner is usually a classifier to decide to use which base learner given the current instance. Instead, the meta learner here estimate the potential predictive error for each base learner with the current sample and the algorithm assigns different weight to the base learners accordingly. The target variable of the meta learning is the predictive error of the base learner e_i, measured by mean absolute error, in our setting. And the dynamic weight for each base learner is $w_i = 1/e_i^2$ and following normalization is required.

Algorithm 3 META-DESR

procedure TRAINMETADESR(D_{tr}, N, α)
  for $i = 1$ to N **do**
    $D_i \leftarrow$ SAMPLEWITHBAGGING(D_{tr}, α); Initialize R_i; TRAIN(R_i, D_i)
  end for
  COMPUTEOUTPUTPROFILES(D_{tr}, R); COMPUTEPREDICTIVEERRORS(D_{tr}, R); Initialize R_{meta}
  for $i = 1$ to $|D_{tr}|$ **do**
    nbr $\leftarrow$ KNEARESTNEIGHBORS(D_{tr_i}, k)
    profNbr $\leftarrow$ MCLOSESTPROFILES(D_{tr_i}, m)
    for $n = 1$ to N **do**
      $\text{pred}_n \leftarrow$ PREDICT(R_n, D_{tr_i})
      COMPUTENEIGHBORERRORS(R_n, nbr, profNbr)
    end for
    COMPUTEDESCRIPTIVESTATS($\{\text{pred}_n\}_{n=1}^{N}$)
  end for
  FITMETALEARNER(R_{meta}); **return** (R, R_{meta})
end procedure

procedure PREDICTMETADESR($D_{te}, R, R_{\text{meta}}$)
  for $i = 1$ to $|D_{te}|$ **do**
    for $n = 1$ to N **do**
      $\text{pred}_n \leftarrow$ PREDICT($R_n, D_{te}[i]$)
    end for
    metaFeatures $\leftarrow$ COMPUTEMETAFEATURES($D_{te}[i], \{\text{pred}_n\}$)
    weights $\leftarrow$ PREDICT(R_{meta}, metaFeatures)
    $\hat{y}_i \leftarrow$ COMBINEPREDICTIONS($\{\text{pred}_n\}$, weights)
  end for
  return $\{\hat{y}_1, \ldots, \hat{y}_{|D_{te}|}\}$
end procedure

Fourth, the dynamic feature-based ensemble strategy is an extension on the feature-based static ensemble. We use a dictionary at the beginning to record the minimal and maximal values of each feature in the training set. Then, in the prediction stage, for each coming instance, the algorithm compares the features and the range values recorded in the dictionary. If any of the feature values are unseen, the algorithm discard it in the predictive modeling by assign a zero weight to the base learners with corresponding features. The idea of the strategy is to avoid using the features which have unseen out-of-range values to mitigate the covariate shift part. One detail that requires extra care is that the number of the learners of the strategy cannot be too small, otherwise for some extreme instances in the testing set where too many features are out-of-bound resulting in no available base regressors. Also, this strategy works better when all independent variables are continuous instead of categorical.

Algorithm 4 Dynamic Feature-Based Ensemble Strategy

procedure TRAINDFBE(D_{tr}, N, β)
  $L \leftarrow [\,],\ H \leftarrow$ COMPUTEFEATURERANGES(D_{tr})
  for $i = 1$ to N **do**
    $f_i \leftarrow$ FEATURERESAMPLE(β)
    $D_i \leftarrow$ SLICE(D_{tr}, f_i)
    Initialize R_i, TRAIN(R_i, D_i), **append** f_i to L
  end for
  return (R, L, H)
end procedure

procedure PREDICTDFBE(D_{te}, R, L, H)
  for $i = 1$ to $|D_{te}|$ **do**
    $x_i \leftarrow D_{te}[i]$, outOfRange $\leftarrow$ CHECKRANGES(x_i, H)
    for $j = 1$ to N **do**
      $f_j \leftarrow L[j],\ \text{weight}_j \leftarrow \begin{cases} 0 & \text{if ISOUTOFRANGE}(f_j, \text{outOfRange}) \\ 1 & \text{otherwise} \end{cases}$
      $\text{pred}_j \leftarrow$ PREDICT($R_j, x_i[f_j]$)
    end for
    NORMALIZEWEIGHTS($\{\text{weight}_j\}_{j=1}^{N}$)
    $\hat{y}_i \leftarrow$ COMBINE($\{\text{pred}_j\}, \{\text{weight}_j\}$)
  end for
  return $\{\hat{y}_1, \ldots, \hat{y}_{|D_{te}|}\}$
end procedure

4 Experiment

4.1 Datasets and Settings

The LGD dataset we use in the study comes from Moody's Default and Recovery Database (DRD). The dataset includes cases of U.S. companies that defaulted with more than $50 million in debt. We merge it with the S&P's Compustat database to match the financial performances of the obligors to the debt instruments when they defaulted. We also obtain macroeconomic data from the website of U.S. Federal Reserve St. Louis to provide information about macroeconomic environment. After merging there are 3,058 samples in the dataset from 715 unique obligors, covering the time period from 1987-04 to 2020-09. Table 1 shows the variable information of our dataset.

Table 1: Variable list

Variable Name	**Description**	**Source**
Profitability	The obligor's profitability ratio	S&P Computstat
LTR	The obligor's long term debt ratio	
DB	The obligor's default barrier	
ReceivableR	The obligor's account receivable ratio	
ORIGINAL_AMNT	Original amount of the debt instrument	Moody's Default and Recovery Database
EFFECTIVE_INTEREST_RATE	The instrument's effective interest rate	
TYPE	Indicating the seniority type of the debt instrument	
RECOVERY_INDUSTRY	The obligor's industry	
CPIAUCSL_PCQ	Quarterly change of CPI index	FRED Economic Data
UNRATE_PCQ	Quarterly change of unemployment rate	
DFF	Federal interest rate	
logR	Quarterly rate of return of S&P 500 Index	
OBLIGOR_ID	The unique ID for obligors (for sorting and splitting purposes)	Moody's Default and Recovery Database
INSTRUMENT_DEFAULT_DATETIME	Indicating the instrument default date-time (for sorting and splitting purposes)	
LGD	Loss given default	

Figure 1 displays the empirical distribution of LGD in our dataset with key descriptive statistics including mean, median, standard deviation and 1st/3rd quartile. The LGD distribution has two modes at 0 (larger) and 1 (smaller). More than 25% percentage of defaulted debt instruments can be fully recovered in history.

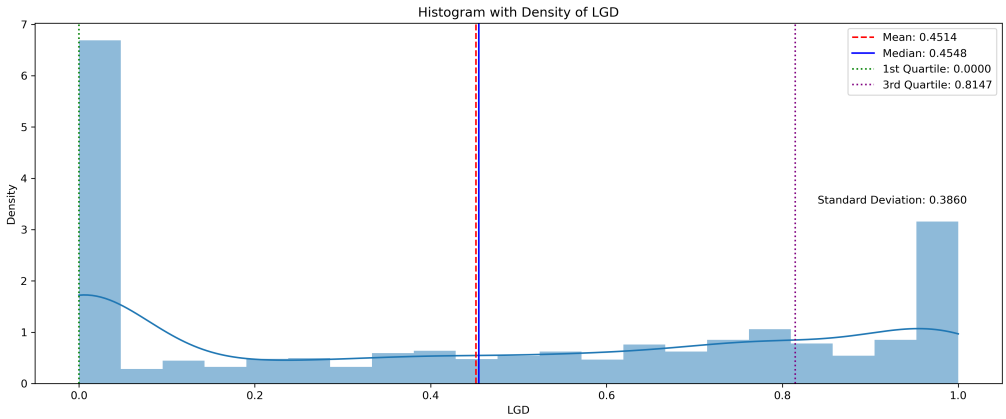

Figure 1: Histogram of LGD with Descriptive Statistics

Here in Figure 2, UMAP (Uniform Manifold Approximation and Projection) is conducted to visualise the spatial information of the independent variables of the LGD dataset (McInnes et al., 2018). We first use principal component analysis to do the dimensionality reduction and select the top 2 principal components. Then, for each year, a UMAP subplot is constructed. From 2, we can observe that the spatial structure of the independent variables of the LGD data varies from year to year. The different colours are for different loss types of the sample. According to the recovery outcomes. We divide all the samples into three categories: fully recovered (labelled as 0), fully lost (labeled as 2), and partially lost (labelled as 1). Some obvious clusters occur in every year, which is consistent with the fact that, in an default event, there can be multiple defaulted debt instrument from the same obligor. Also, by looking at the UMAP plots for different years, the distributions of the data points in each year are quite different, supporting the statement that LGD data may present strong dataset shift.

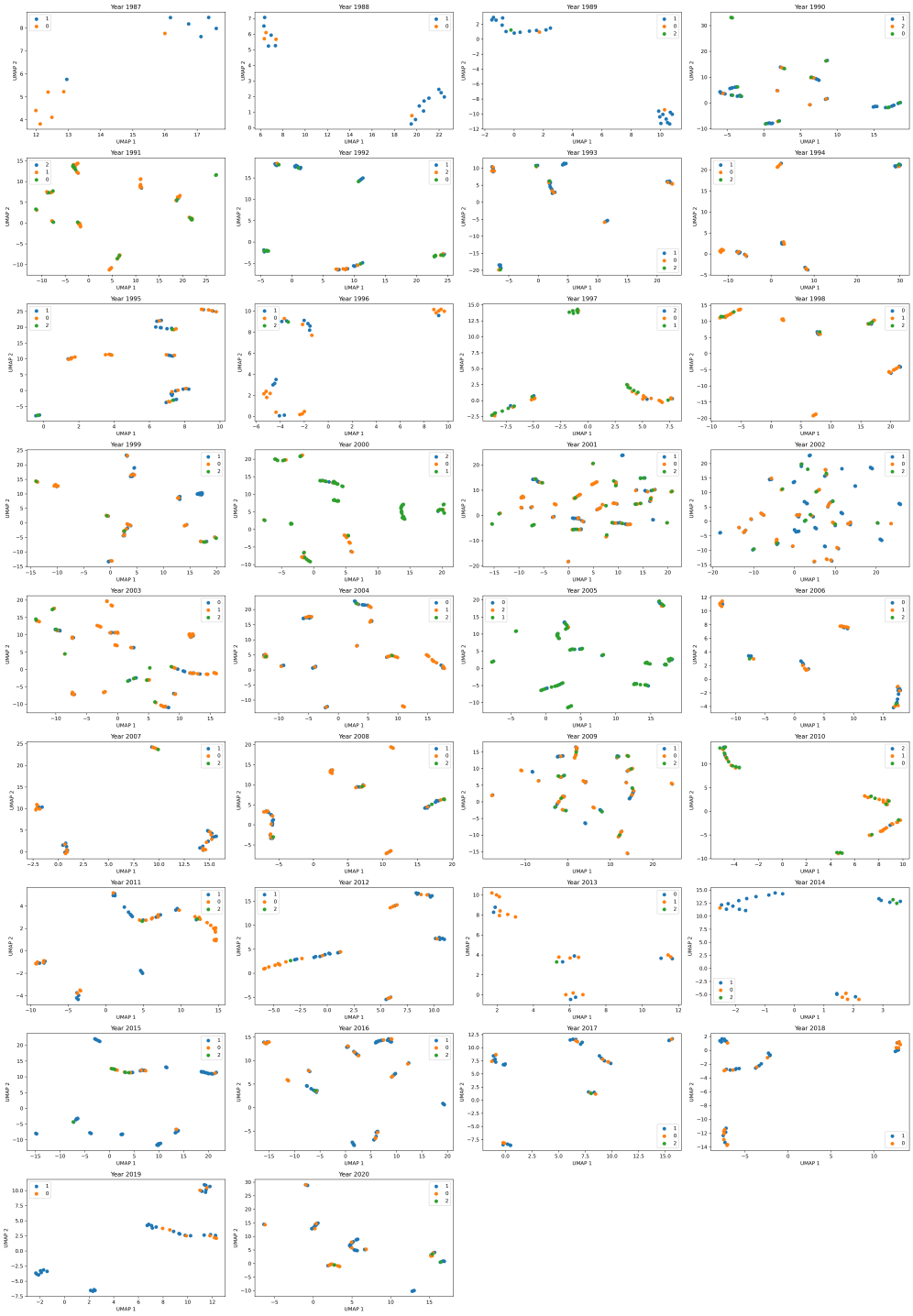

Figure 2: UMAP Visualisation for LGD data points in each year

4.2 Behaviours of Various Regressors in OOT

First, in this section, we choose multiple commonly used machine learning models to examine its generalization ability to the testing set with OOT setting, including linear regression (LR), support vector regressor (SVR), LightGBM, 4-layer Deep neural network (DNN), random forest (RF), and K-Neareset—Neighbors (KNN). LGD is a continuous variable falling between 0 and 1, inclusively, with a multi-modal distribution. The modes of the LGD are typically 0 (fully recovered) and 1 (fully lost). These characteristics makes the machine learning models difficult to closely fit the data and make reasonable predictions within the bound. We initially train the models with the training set, then let them make predictions on both training set and testing set. By comparing the statistics and the scatter plot between the predicted values and the true values on both sets, we can analyse the behaviours of different regressors and understand the temporal properties of the LGD data better.

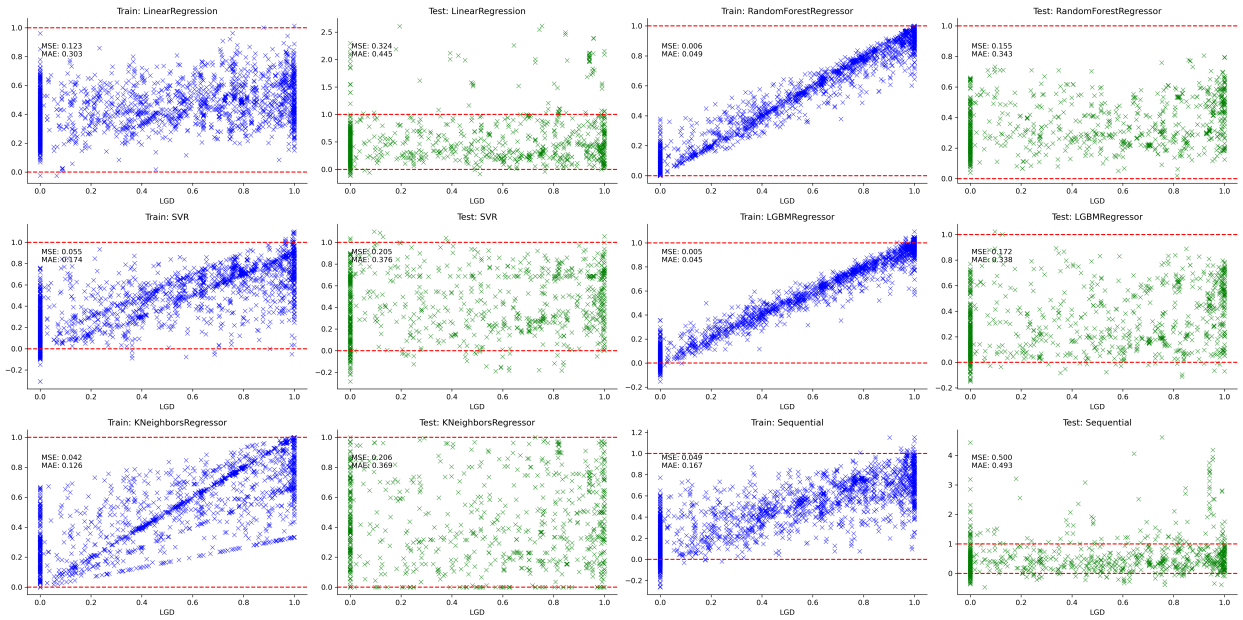

Figure 3: Different Regressors' Behaviours on Out-of-time Split

Figure 3 shows the scatter plot of the predicted the values against the actual LGD, which means if the pattern of the dots fits closer to the function $y = x$, the model has a better performance. The plots with blue crosses are for training set prediction while the plots with green crosses are for testing set prediction. The predictive errors, MAE and MSE, are annotated in the plots. The horizontal red dotted reference lines in each subplot indicate the theoretical lower and upper bound of the predicted LGD values (0 and 1). But as what are displayed in the plots, some models' outputs exceed it significantly. In general, almost all

machine learning models learn the distribution of the LGD and make the bound of the output on training set within $[0, 1]$. However, with the OOT setting, the performance of the models deteriorate significantly, with two common problems: (1) the predicted distribution deviates from the true distribution massively; and (2) the output cannot be bounded within 0 and 1, in some extreme cases such as linear regression and neural network, there are some predictions with value larger than 2. On the other hand, random forest is relatively conservative in predicting unseen samples, as the predictions on testing set of it rarely exceed 0.8.

Table 2: The descriptive statistics and predictive errors of different models on both sets

	Predictions on Training Set							**Predictions on Testing Set**						
Model Name	**Mean**	**Std**	**Median**	**Min**	**Max**	**MSE**	**MAE**	**Mean**	**Std**	**Median**	**Min**	**Max**	**MSE**	**MAE**
Linear Regression	0.4472	0.1565	0.4333	-0.0248	1.0127	0.1227	0.3026	0.5373	0.4674	0.4135	-0.1153	2.6130	0.3238	0.4451
SVR	0.4422	0.2853	0.4544	-0.3075	1.1001	0.0554	0.1739	0.4354	0.2882	0.4317	-0.2835	1.0989	0.2049	0.3759
Random Forest	0.4478	0.3441	0.4466	0	1	0.0059	0.0492	0.3473	0.1595	0.3077	0.0192	0.8062	0.1552	0.3432
LightGBM	0.4472	0.3594	0.4514	-0.1524	1.0934	0.0046	0.0446	0.3141	0.2401	0.2590	-0.1492	1.0228	0.1719	0.3882
KNN	0.4418	0.3165	0.4485	0	1	0.0415	0.1259	0.3894	0.2874	0.3469	0	1	0.2059	0.3687
Deep Neural Network	0.4365	0.2853	0.4482	-0.2615	1.1529	0.0486	0.1669	0.5592	0.6574	0.4071	-0.4669	4.6091	0.4999	0.4934

In practice, to avoid the prediction deviate from the interval $[0, 1]$, clipping or some transformations are made to ensure the predictions are in bound. However, this effort has only marginal effect on the final predictive accuracy, as we can find in the result displayed in Table 2, the parameterised models (linear regression and deep neural network) are more susceptible to the out of range problem and the predictive standard deviations are higher, while the problems of others are not so serious. Another notable pattern is that, compared with mean LGD of the testing set (0.46), the parametric model tends to overestimate the LGD while the non-parametric model are inclined to underestimate the LGD.

4.3 Detecting and Monitoring the Dataset Shift

We firstly conduct shift detection on the target variable LGD. Figure 4 displays the density plot of the LGD in both training set and testing set, indicating no significant change in distribution. The KS statistic is 0.0494 with p-value of 0.0833 (please refer to Table 3), implying the accept the null hypothesis of two data series are from the same distribution. In the first regime, there are several events related to the outburst of corporate bond default and potentially high loss given default including 2002 Enron scandal and 2008-09 financial crisis. But the distribution of LGD in the second regime does not present significant difference, except the proportion of fully recovered samples is slightly higher in the training period than testing period. Table 3 shows the test results for each feature in the LGD dataset and the overall input vectors. The asterisk (*)

next to the p-values indicates the p-value is significant with Bonferroni correction, the threshold of which is about 0.0008.

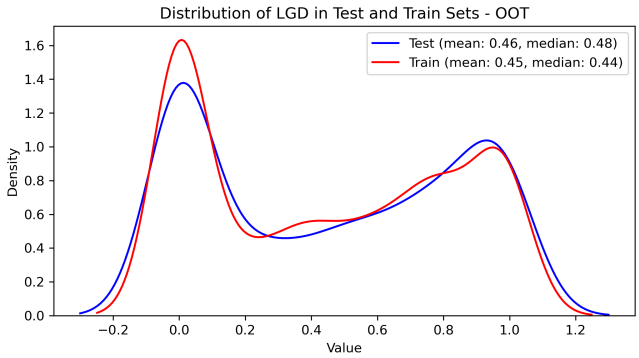

Figure 4: The distributions of LGD in Training and Testing Sets - Out-of-time Split

Table 3: Dataset Shift Detection Result for OOT and OOS

		Out-of-time		**Out-of-sample**	
Test Name	**Variable Name**	**Statistics**	**p-value**	**Statistics**	**p-value**
KS Test	Profitability	0.1859	1.053e-19*	0.0373	0.3230
	LTR	0.1116	1.963e-07*	0.0347	0.4097
	DB	0.1458	2.241e-12*	0.0232	0.8681
	ReceivableR	0.1800	1.157e-18*	0.0209	0.9337
	ORIGINAL_AMNT	0.3972	3.453e-9*	0.0205	0.9442
	EFFECTIVE_INTEREST_RATE	0.2316	1.035e-30*	0.0536	0.0477
	TYPE	0.1719	4.725e-17*	0.0241	0.8379
	RECOVERY_INDUSTRY	0.2187	2.103e-27*	0.0233	0.8676
	DFF	0.8248	5.241e-322*	0.0261	0.7599
	UNRATE_PCQ	0.3210	4.040e-59*	0.0234	0.8649
	CPIAUCSL_PCQ	0.1874	3.294e-20*	0.0230	0.8752
	logR	0.1468	1.494e-12*	0.0386	0.2858
	LGD	0.0494	0.0833	0.0403	0.2398
MMD Test	All independent variables	0.0053		0.0014	

From Table 3, we notice that, in the out-of-time split, all features has changed significantly in distribution from the training period to the testing period. The financial ratios are more stable, while the macroeconomic variables have the largest change between different regimes. Particularly, the federal interest rate (variable **DFF**) has the highest discrepancy, which also reflects the reality that interest environment has changed dramatically over the last decades. The previous distribution of the federal interest rate can be tricky to exploited for today's modelling work as the input. Other three macroeconomic variables are changes differentiated quarterly instead of the absolute values. However the one related to the unemployment rate is not very stable (0.3210). The debt-level characteristics also experience the moderate shift, among of which the original loan amount is on top (0.3972). The MMD test we conduct is with RBF kernel and mixed gamma parameter setting $[0.001, 0.01, 0.1, 0.5]$ to have a good balance between capturing the subtlety of the change in the data and focusing the high-level shift (Ouyang & Key 2021). The statistics of MMD test on OOT split is 0.0053. For comparison, the same test on the OOS split is 0.0014.

We also include the test result for OOS split in the Table 3, revealing that, including LGD, all variables in the KS test are with the statistics between 0.02 and 0.06. Besides the p-values of the test with OOS split indicate all tests should accept the null hypothesis which there is no significant distributional change between two parts of the data. In general, the LGD dataset presents very strong covariate shift problem, as the LGD's distribution in the two different regimes are not significantly different but the explanatory variables are changing all the time.

After having a snapshot of how the data has different distribution in the training set and the testing set. We extend the single time detection of the distributional shift to a continuous monitoring setting. Monitoring the samples' distributional shift has been widely recognized and adopted in the practice of credit scoring (Thomas et al. 2017). However, regarding LGD modeling practices, it did not draw people's attention that much.

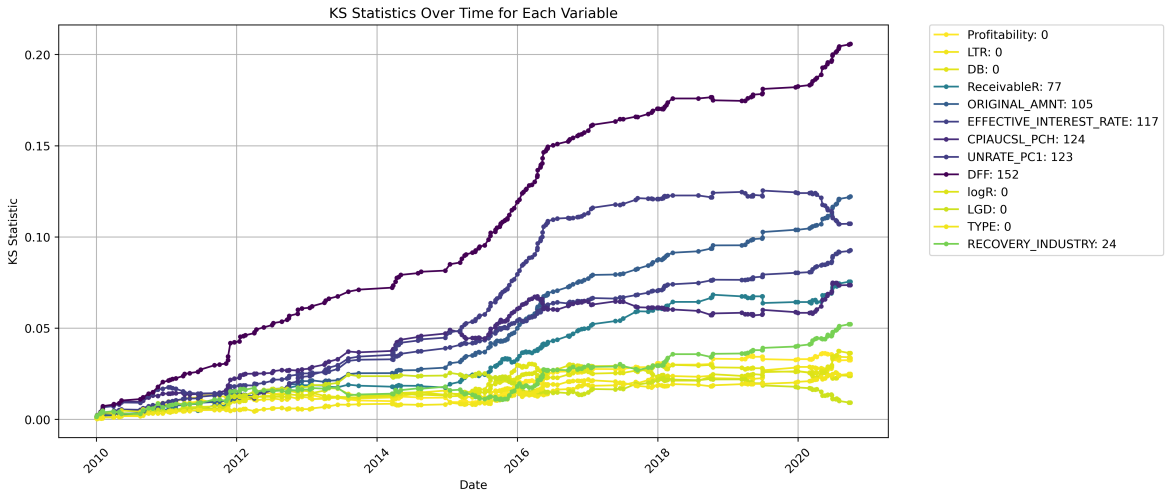

Figure 5: Monitoring the Characteristic Distributional Shift

We use a sliding window technique to continuously monitoring the distributional shift of each characteristic as well as the target variable, ultimate LGD. To be consistent with the our machine learning model development, we choose the whole training set as the reference window, also the initial status of the sliding window. Then, we treat each day with coming samples as one time step. The window is rolled forward with new samples from the next available date added and old samples from the oldest date in current window frame removed. The testing set covers 183 days with LGD data over the 10 years. Figure 5 displays how the distributions of different variables in the dataset deviates from that of reference window. The x-axis indicates the closing time of the current sliding window, and the y-axis is for the KS statistics. The colours of the lines indicates how frequent the associated variable has experienced distributional change. The more times a variable has significant shifts, the darker the line is. The specific numbers of how many times the KS-tests are significant can be find in the legend. The variable DFF are associated the most times of significant shifts (152 out of 183, approximately 83% of the test time). While variables Profitability, LTR, DB, logR (the quarterly return of S&P 500 index), the target variable LGD, and the type of the debt instrument does not has significant changes over the testing time period at all. The result shows the financial ratios are relatively more stable while the macroeconomic variables changes drastically, which is consistent with the findings in the previous single time train-test distribution comparison. In our analysis, in order to have a good comparison and be able to monitor the distribution evolve continuously, we transform the categorical variables into a dense numerical feature with autoencoder and use KS test on all variables. But for the sake of intepretability in the real business scenario, alternatively, we can bin the continuous variable and analyse all categorised variables with stability index (Thomas et al. 2017).

4.4 Ensemble Learning

In this section, a comprehensive comparison study for different base models and different ensemble learning techniques will be conducted on LGD modelling. We first test the modelling performance of single models, covering Linear Regression (LR), Support Vector Regressor (SVR), K-Neareset-Neighbors (KNN), Deep Neural Network (DNN), Random Forest (RF), and LightGBM (LGBM). Theoretically, RF and LGBM are already ensemble learning techniques, but considering the practical situation, we regard them as single base models as well. To make the experiment result meaningful and robust, we use 5-fold chronological cross validation on all data achieved by Python Scikit-Learn's *tscv* function and report the mean performance for each model, ensuring in each trial of the experiment the testing data are observed after the training data. The evaluation metric we use in the experiment are mean squared error (MSE). One more technical detail here is that we manually limit the output of the models bounded within range $[0, 1]$ by clipping the values (for DNN models, we use sigmoid activation function on the output layer instead of clipping, since our previous experiments indicates that, for neural network, sigmoid activation function works better than simply truncate the output values as the deep neural network can learn the distribution better from the backpropagation) to make the output more meaningful and further lower the predictive errors.

The first row of Table 4 displays the performance of each single model on LGD modeling. RF and LGBM outperform other techniques significantly, followed by DNN. And LR achieves 0.1581 MSE on the 5-fold time series split, better than SVR and KNN. A simpler relationship learnt from the training set may have the better generalization ability. The second part of Table 4 is the performances of static ensemble learning methods consisting one or multiple types of base models. To make the comparison meaningful, we keep the hyperparameter setting of the base models same as the one examined in the single model test. Also, as the ensemble learning performance can be unstable, for the ensemble learning techniques with tunable hyperparameters, such as the bagging ratio or the number of base learners, we conduct 100 trials of different hyperparameter combinations and report the average performance. The first one is heterogeneous ensemble consisting of all each base learners, achieving the 0.1273 MSE, but this performance is worse than the best base model (LGBM). Then we test the simple average ensemble with bagging on each base learner. In general, this ensemble elevates the modeling performance of using a single base learner, with one exception of DNN. Compared with other base learners, DNN does not work very well in ensemble learning setup. We infer that using smaller subset of the data to train a deep learning may cause stronger overfitting problem. We notice that the ensemble learning techniques can even work on top of the ensemble learning models (e.g., RF and LGBM), especially when the high-level ensemble strategy provides different source

of diversity. Among the static ensemble learning strategies, feature-based static model is significantly better than the other two counterparts.

Table 4: The Performances of Different Ensemble Modelling Techniques (MSE)

	LR	**SVR**	**KNN**	**DNN**	**RF**	**LGBM**
Single Model	0.1581	0.1667	0.1906	0.1414	0.1226	0.1216
Heterogeneous Ensemble			0.1273			
Bagging Static	0.1570	0.1620	0.1622	0.1481	0.1189	0.1185
Feature-Based Static	0.1453	0.1416	0.1393	0.1392	0.1171	0.1179
Bagging Distance-Based Dynamic	0.1543	0.1578	0.1610	0.1480	0.1165	0.1156
K-Means Distance-Based Dynamic	0.1606	0.1611	0.1719	0.1524	0.1230	0.1174
META-DESR	0.1555	0.1548	0.1619	0.1408	0.1145	0.1147
Feature-Based Dynamic	**0.1425**	**0.1381**	**0.1356**	**0.1311**	**0.1139**	**0.1118**

Table 5: The Performances of Different Ensemble Modelling Techniques (MAE)

	LR	**SVR**	**KNN**	**DNN**	**RF**	**LGBM**
Single Model	0.4438	0.3767	0.3682	0.4957	0.3419	0.3861
Heterogeneous Ensemble			0.3472			
Bagging Static	0.3983	0.3625	0.3668	0.4117	0.3284	0.3229
Feature-Based Static	0.3891	0.3538	0.3564	0.3931	0.3163	0.3192
Bagging Distance-Based Dynamic	0.3926	0.3549	0.3615	0.3994	0.3112	0.3168
K-Means Distance-Based Dynamic	0.4068	0.3691	0.3728	0.4045	0.3217	0.3274
META-DESR	0.3847	0.3462	0.3537	0.3886	0.3069	0.3081
Feature-Based Dynamic	**0.3728**	**0.3364**	**0.3387**	**0.3795**	**0.3024**	**0.3063**

The third part of the Table 4 contains the performances of four dynamic ensemble learning strategies with different base learners. The result shows that the K-Means distance-based dynamic ensemble does not perform well compared with other ensemble techniques. Because the number of clusters are predetermined

and the found clusters with the training set may not consistent with the patterns in the testing set (see Figure 2 for some supporting evidence of evolving clusters of LGD data). In this case, the ensemble algorithm cannot assign the most suitable base learner with higher weights and the performance of the ensemble can be even worse than the static simple averaging. The bagging distance-based dynamic ensemble works better for all base models than that. The META-DESR, although with more advanced technical design and requiring more computation, only brings slight improvement compared to other bagging based ensemble strategies. Among all dynamic ensemble learning strategies, feature-based dynamic ensemble is the best, which proves the rationale behind the strategy is effective. Although we cannot avoid the change of the relationship between the independent variables and the target variable in different regimes, by avoiding use the out-of-range features, the mitigation of the covariate shift, which is the main type of the dataset shift in LGD dataset, is almost guaranteed. Table 5 presents the model performance comparison in terms of MAE. The result remains the same. In general, we can also find the feature-based ensembles work better than sample-based ensembles, which is consistent with the insights we obtained from the previous exploration on the LGD data that it is relatively hard to find time-invariant clusters but some of the features are relatively stable over the time.

5 Conclusion

This paper explores the effectiveness of dynamic ensemble learning dealing with the dataset shift problem in regression tasks. In real-world deployment, machine learning solutions require frequent recalibration or retraining with new data. This study provides a complete framework to combat the dataset shift problem in regression tasks, from the initial detection and problem visualization, to find a proper dynamic ensemble integration method accordingly to mitigate the issue, which can effectively reduce the model risk and lower the maintenance cost. Besides, continuous data monitoring is suggested over periodically examination to make sure the in-time detection of the potential shifts. In the experiment section, we explore the dataset shift problem of LGD modeling by examining the distribution of the variables in different regimes statistically, analyze why single model's performance can be deteriorate massively in the testing set, and use multiple novel dynamic ensemble learning techniques to alleviate the problem and further improve the modeling accuracy of the commonly used machine learning models. Here are the key takeaways from our experiment:

1. LGD modeling can be difficult because there is strong dataset shift problem. The distribution of LGD has not undergone significant change, however, due to the change of the macroeconomic environment

and some extreme financial events happened, the independent variables, including the economic indicators, corporates' financial performances and the debt-level features have been changing all the time.

2. Compared with using a single model, ensemble learning strategies are more robust and can effectively mitigate the dataset shift problem. Even for those commonly used ensemble learning models such as RF and boosting models, proper ensemble learning techniques can further improve the performance. We explore different ensemble strategies by making the base learners either 'horizontally diverse' (sample-based resampling) or 'vertically' (feature-based resampling) and find out in LGD modeling the latter strategy works better, which is also consistent with our findings that some features are more time-invariant while the others are not.

3. Lastly, compared with static ensemble selection or simple average ensemble learning, dynamic ensemble selection can help improve the modeling performance by allocating the best base learners higher weights and reducing the weights assigned to potentially weak learners, on an instance basis.

However, the dynamic ensemble approach has its own limits, since it typically involve more computational overhead than static ensembles. Each new instance requires a real-time selection of the most appropriate learners, which can be computationally intensive, especially with large datasets or complex models. Also the performance fluctuation of dynamic ensemble can be higher than those static methods or single model methods. In the future work, we will keep exploring and improve the dynamic ensemble method in stated two ways by optimizing the computation complexity and related data structures. Also, for LGD modeling, we will keep focusing on exploiting new data to explain it with more robust models to make the prediction more accurate.

References

Anderson, E. & Cheng, A.-r. (2022), 'Portfolio choices with many big models', *Management Science* **68**(1), 690–715.

Armstrong, R. A. (2014), 'When to use the b onferroni correction', *Ophthalmic and Physiological Optics* **34**(5), 502–508.

Blanchet, J., Chen, L. & Zhou, X. Y. (2022), 'Distributionally robust mean-variance portfolio selection with wasserstein distances', *Management Science* **68**(9), 6382–6410.

Bolón-Canedo, V., Sánchez-Maroño, N. & Alonso-Betanzos, A. (2016), 'Feature selection for high-dimensional data', *Progress in Artificial Intelligence* **5**, 65–75.

Breiman, L. (2001), 'Random forests', *Machine learning* **45**, 5–32.

Britto Jr, A. S., Sabourin, R. & Oliveira, L. E. (2014), 'Dynamic selection of classifiers—a comprehensive review', *Pattern recognition* **47**(11), 3665–3680.

Chen, T. & Guestrin, C. (2016), Xgboost: A scalable tree boosting system, *in* 'Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining', pp. 785–794.

Cruz, R. M., Hafemann, L. G., Sabourin, R. & Cavalcanti, G. D. (2020), 'Deslib: A dynamic ensemble selection library in python', *The Journal of Machine Learning Research* **21**(1), 283–287.

Cruz, R. M., Sabourin, R. & Cavalcanti, G. D. (2018), 'Dynamic classifier selection: Recent advances and perspectives', *Information Fusion* **41**, 195–216.

Cruz, R. M., Sabourin, R., Cavalcanti, G. D. & Ren, T. I. (2015), 'Meta-des: A dynamic ensemble selection framework using meta-learning', *Pattern recognition* **48**(5), 1925–1935.

Du, L., Gao, R., Suganthan, P. N. & Wang, D. Z. (2022), 'Bayesian optimization based dynamic ensemble for time series forecasting', *Information Sciences* **591**, 155–175.

Dubiel-Teleszynski, T., Kalogeropoulos, K. & Karouzakis, N. (2024), 'Sequential learning and economic benefits from dynamic term structure models', *Management Science* **70**(4), 2236–2254.

Fang, T., Lu, N., Niu, G. & Sugiyama, M. (2020), 'Rethinking importance weighting for deep learning under distribution shift', *Advances in neural information processing systems* **33**, 11996–12007.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B. & Smola, A. (2012), 'A kernel two-sample test', *The Journal of Machine Learning Research* **13**(1), 723–773.

Guo, C., Liu, M. & Lu, M. (2021), 'A dynamic ensemble learning algorithm based on k-means for icu mortality prediction', *Applied Soft Computing* **103**, 107166.

Hou, W.-h., Wang, X.-k., Zhang, H.-y., Wang, J.-q. & Li, L. (2020), 'A novel dynamic ensemble selection classifier for an imbalanced data set: An application for credit risk assessment', *Knowledge-Based Systems* **208**, 106462.

Kalotay, E. A. & Altman, E. I. (2017), 'Intertemporal forecasts of defaulted bond recoveries and portfolio losses', *Review of Finance* **21**(1), 433–463.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q. & Liu, T.-Y. (2017), 'Lightgbm: A highly efficient gradient boosting decision tree', *Advances in neural information processing systems* **30**.

Li, F., Lam, H. & Prusty, S. (2020), Robust importance weighting for covariate shift, *in* 'International conference on artificial intelligence and statistics', PMLR, pp. 352–362.

Lipton, Z., Wang, Y.-X. & Smola, A. (2018), Detecting and correcting for label shift with black box predictors, *in* 'International conference on machine learning', PMLR, pp. 3122–3130.

McInnes, L., Healy, J. & Melville, J. (2018), 'Umap: Uniform manifold approximation and projection for dimension reduction', *arXiv preprint arXiv:1802.03426* .

Moreno-Torres, J. G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N. V. & Herrera, F. (2012), 'A unifying view on dataset shift in classification', *Pattern recognition* **45**(1), 521–530.

Nazemi, A., Baumann, F. & Fabozzi, F. J. (2022), 'Intertemporal defaulted bond recoveries prediction via machine learning', *European Journal of Operational Research* **297**(3), 1162–1177.

Ouyang, L. & Key, A. (2021), 'Maximum mean discrepancy for generalization in the presence of distribution and missingness shift', *arXiv preprint arXiv:2111.10344* .

Qi, M., Cao, Y. & Shen, Z.-J. (2022), 'Distributionally robust conditional quantile prediction with fixed design', *Management Science* **68**(3), 1639–1658.

Quinonero-Candela, J., Sugiyama, M., Schwaighofer, A. & Lawrence, N. D. (2008), *Dataset shift in machine learning*, Mit Press.

Rabanser, S., Günnemann, S. & Lipton, Z. (2019), 'Failing loudly: An empirical study of methods for detecting dataset shift', *Advances in Neural Information Processing Systems* **32**.

Saadallah, A. & Morik, K. (2021), Online ensemble aggregation using deep reinforcement learning for time series forecasting, *in* '2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA)', IEEE, pp. 1–8.

Sagi, O. & Rokach, L. (2018), 'Ensemble learning: A survey', *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **8**(4), e1249.

Shimodaira, H. (2000), 'Improving predictive inference under covariate shift by weighting the log-likelihood function', *Journal of statistical planning and inference* **90**(2), 227–244.

Si, N., Zhang, F., Zhou, Z. & Blanchet, J. (2023), 'Distributionally robust batch contextual bandits', *Management Science* **69**(10), 5772–5793.

Simester, D., Timoshenko, A. & Zoumpoulis, S. I. (2020), 'Targeting prospective customers: Robustness of machine-learning methods to typical data challenges', *Management Science* **66**(6), 2495–2522.

Sugiyama, M. & Kawanabe, M. (2012), *Machine learning in non-stationary environments: Introduction to covariate shift adaptation*, MIT press.

Sugiyama, M., Krauledat, M. & Müller, K.-R. (2007), 'Covariate shift adaptation by importance weighted cross validation.', *Journal of Machine Learning Research* **8**(5).

Thomas, L., Crook, J. & Edelman, D. (2017), *Credit scoring and its applications*, SIAM.

Tian, Y. & Zhang, Y. (2022), 'A comprehensive survey on regularization strategies in machine learning', *Information Fusion* **80**, 146–166.

Tschannen, M., Bachem, O. & Lucic, M. (2018), 'Recent advances in autoencoder-based representation learning', *arXiv preprint arXiv:1812.05069* .

Yu, X., Wu, W., Liao, X. & Han, Y. (2023), 'Dynamic stock-decision ensemble strategy based on deep reinforcement learning', *Applied Intelligence* **53**(2), 2452–2470.