



UNIVERSITÀ  
DEGLI STUDI  
DI TRIESTE

# DISTRIBUTION SHIFT

A Study on Their Effects on Statistical Models and  
Strategies for Mitigation

---

Andrea Spinelli, Giacomo Amerio,  
Giovanni Lucarelli, Tommaso Piscitelli

University of Trieste

# Table of contents

---

1. Introduction
2. Data Generation
3. Performance Degradation
4. Performance Enhancement
5. Conclusion

# Introduction

---

## Dataset shift

- **Dataset shift** is a common problem in machine learning.
- It occurs when the distribution of the training data differs from the distribution of the test data.
- This can lead to a decrease in the performance of the model.

The two most common and well-studied causes of dataset shift are:

- Sample selection bias
- non stationary environments

## Aims of project

This project aims to evaluate the impact of simple **Covariate shift** in the input distribution on the performance of robust models within the context of a synthetic binary classification task.

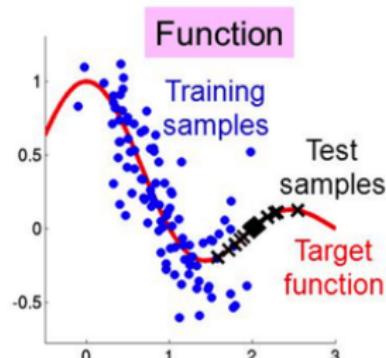
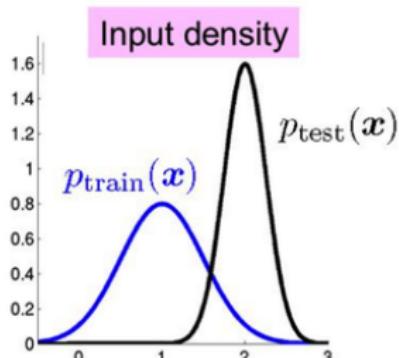
Key questions addressed in this study include:

- How do different types of covariate shifts affect the performance of robust models?
- Are certain models inherently more robust to simple covariate shifts?
- What strategies can be employed to improve model performance following such shifts?

# Covariate shift

Can be formally defined as follows. Consider an input variable  $X$  and a response variable  $Y$ , where  $X \rightarrow Y$  represents the relationship between the two. Let  $P_{\text{tra}}$  denote the probability distribution of the training data and  $P_{\text{tst}}$  denote the probability distribution of the test data. A covariate shift occurs when:

$$P_{\text{tra}}(Y | X) = P_{\text{tst}}(Y | X) \quad \text{but} \quad P_{\text{tra}}(X) \neq P_{\text{tst}}(X).$$



# Example

Consider a model designed to distinguish between cats and dogs:

## Training set:



## Test set:



- Model will not accurately distinguish between cats and dogs because the feature distribution will differ.
- Changes in the input distribution can significantly impact the model's accuracy.

## Inaccurate model

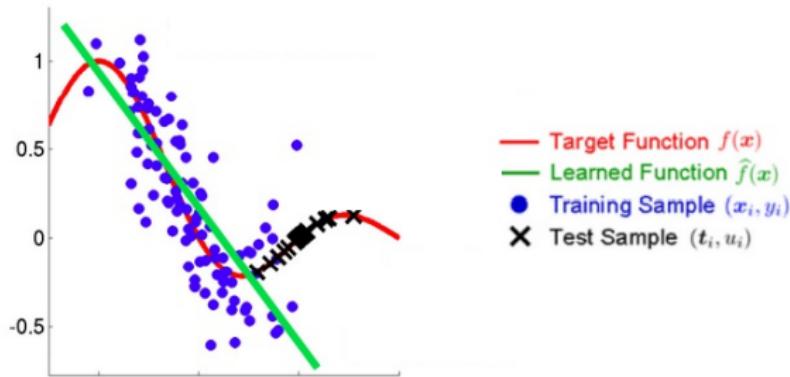


Figure 1: Example of inaccurate model.

In this study, we analyze the effects of distribution shift on different statistical models and propose strategies for its mitigation.

## Data Generation

---

## Training Dataset: Features

---

The dataset consists of  $n = 10^4$  observations with 3 features and 1 binary target variable.

## Training Dataset: Features

The dataset consists of  $n = 10^4$  observations with 3 features and 1 binary target variable.

Features:

- $X_{\text{train}} = (X_{\text{train}1}, X_{\text{train}2}, X_{\text{train}3}) \sim \mathcal{N}(\boldsymbol{\mu}_{\text{train}}, \boldsymbol{\Sigma}_{\text{train}})$
- $\mu_{\text{train}i} \sim \mathcal{U}_{[0,1]}$  for  $i = 1, 2, 3$
- $[\boldsymbol{\Sigma}_{\text{train}}]_{i,j} \sim \mathcal{U}_{[-1,1]}$  for  $i, j = 1, 2, 3$

Note: The  $\boldsymbol{\Sigma}$  randomly generated has been transformed to a symmetric and positive semidefinite matrix by computing  $\boldsymbol{\Sigma}\boldsymbol{\Sigma}^T$ .

## Training Dataset: Target Variable

Building the **target variable**  $Y \in \{0, 1\}$ :

1.

$$z = \beta_0 + \sum_{i=1}^3 \beta_i x_i + \sum_{i=1}^3 \beta_{ii} x_i^2 + \sum_{i=1}^2 \sum_{j=i+1}^3 \beta_{ij} x_i x_j, \quad \beta. \sim \mathcal{U}_{[-1,1]}$$

2.

$$p = \frac{1}{1 + e^{-z}}$$

3.

$$Y \sim \text{Be}(p)$$

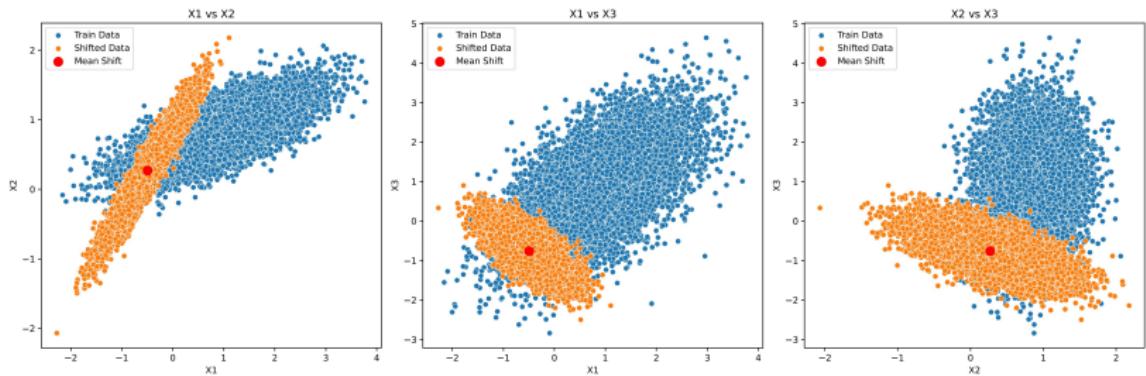
## Testing Dataset

---

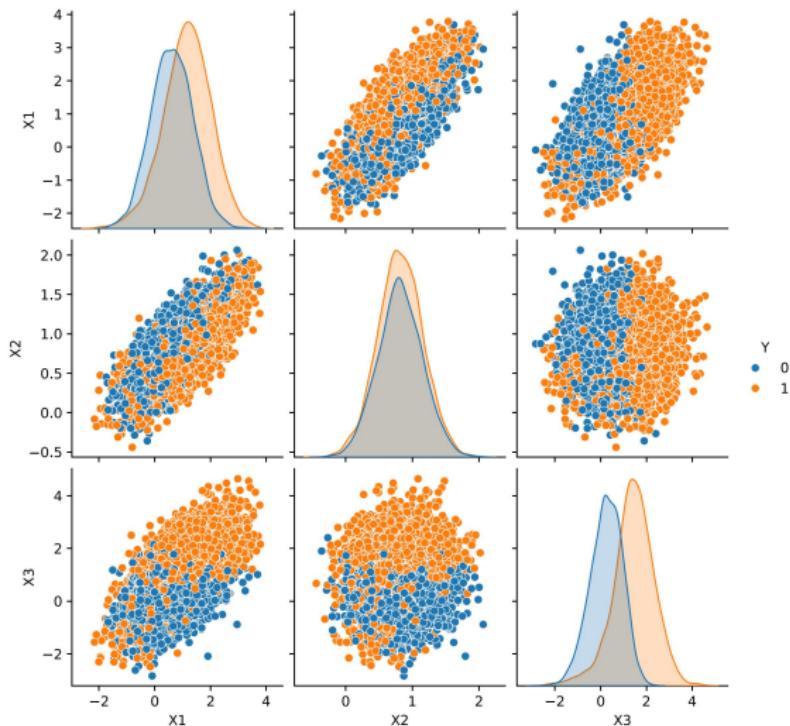
Same dataset structure as the train set, but:

- $X_{\text{shift}} = (X_{\text{shift}1}, X_{\text{shift}2}, X_{\text{shift}3}) \sim \mathcal{N}(\boldsymbol{\mu}_{\text{shift}}, \boldsymbol{\Sigma}_{\text{shift}})$
- $\boldsymbol{\mu}_{\text{shift}} = Q_{0.05}(X_{\text{train}})$
- $[\boldsymbol{\Sigma}_{\text{shift}}]_{i,j} \sim \mathcal{U}_{[-0.5, 0.5]}$  for  $i, j = 1, 2, 3$

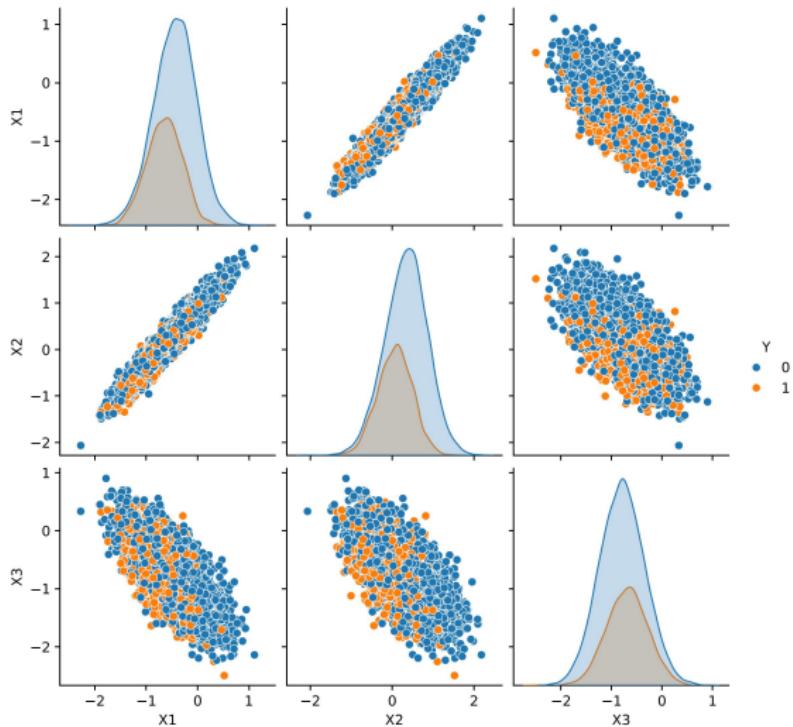
# Original and Shifted Features



# Label Distribution in Train Set



# Label Distribution in Shifted Test Set



Note: IR from 1.19 to 2.36

## Testing Mixture

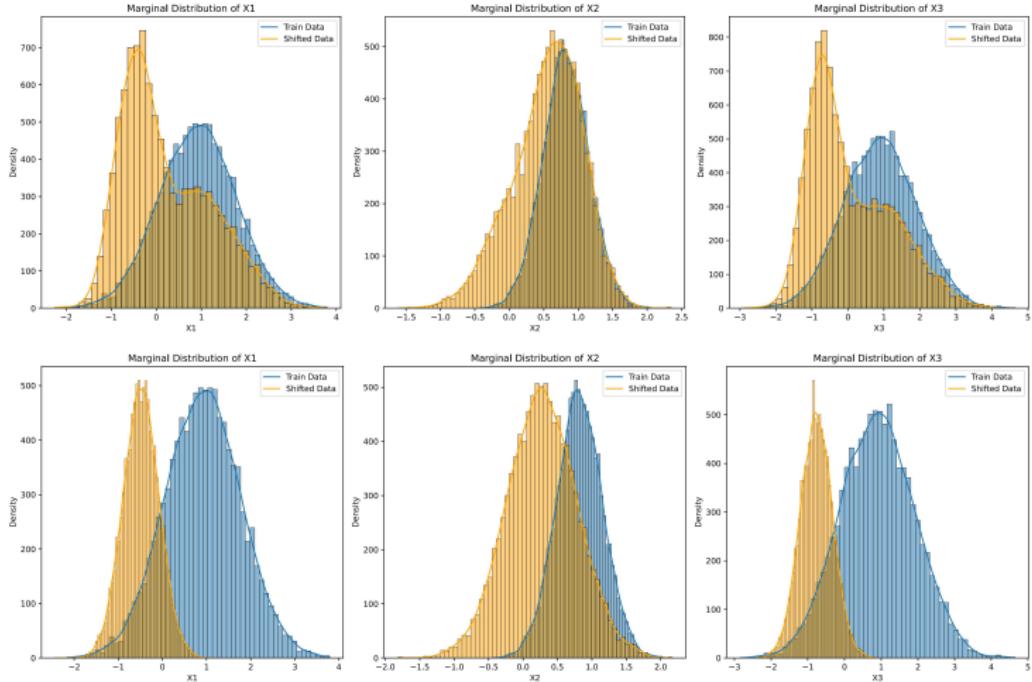
Series of datasets using **statistical mixtures** of the training features distribution and the fully shifted distribution.

$$X_\alpha \sim \alpha \cdot \mathcal{N}(\boldsymbol{\mu}_{\text{shift}}, \boldsymbol{\Sigma}_{\text{shift}}) + (1 - \alpha) \cdot \mathcal{N}(\boldsymbol{\mu}_{\text{train}}, \boldsymbol{\Sigma}_{\text{train}})$$

$$\alpha \in \{0.0, 0.1, \dots, 1.0\}$$

$Y_\alpha$  generated as before

Note:  $X_{0.0}$  and  $X_{\text{train}}$  come from the same distribution, but the former are used as fresh new data.



Top:  $\alpha = 0.5$ . Bottom:  $\alpha = 1.0$ .

## Performance Degradation

---

# Performance Degradation

## Models

- Random Forest
- Gradient Boosting
- XGBoost
- Logistic Regression [baseline]

## Performance Metric

We used the **Area Under the Receiver Operating Characteristic Curve (ROC-AUC)** as the performance metric for our models.

## Fine Tuning

We performed a **hyperparameter tuning** to optimise the performance of our models. To do this, we used the **Grid Search** method with 5-fold cross-validation.

# Logistic Regression

---

**Logistic Regression** is a simple linear model used for binary classification.

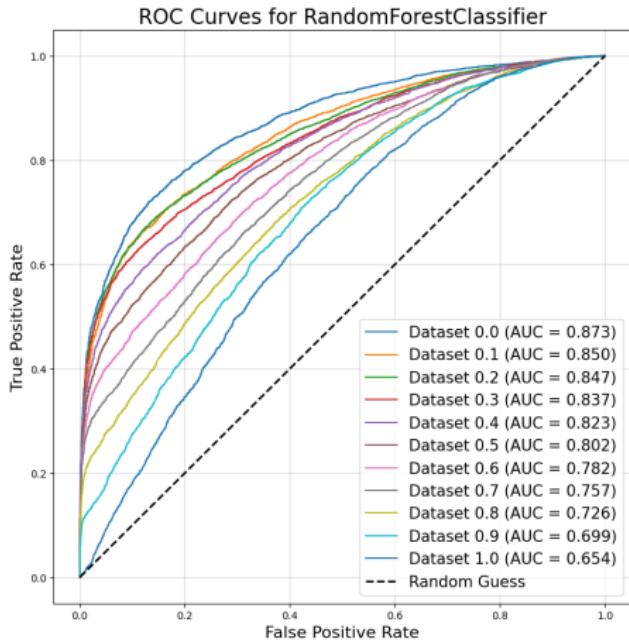
TODO

# Random Forests

**Random Forest** is an ensemble learning method that builds multiple decision trees during training.

It outputs the class that is the majority vote of the individual trees.

Hyperparameter	Value
<i>n_estimators</i>	125
<i>criterion</i>	<i>gini</i>
<i>max_depth</i>	5
<i>min_samples_split</i>	5
<i>min_samples_leaf</i>	1



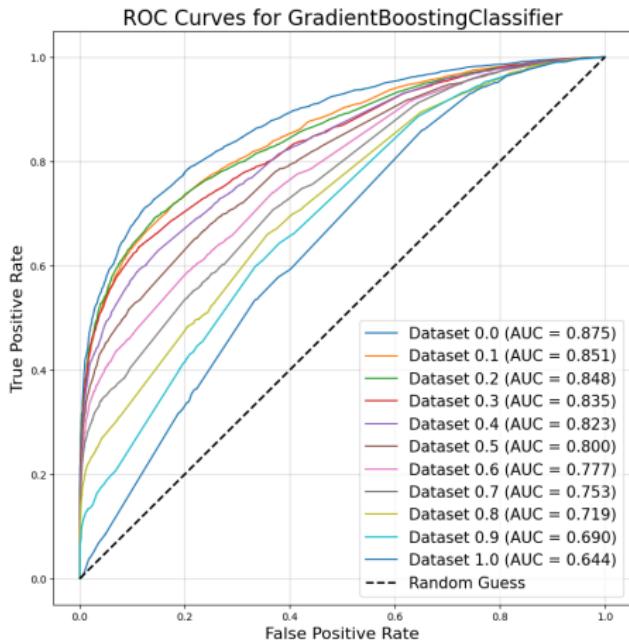
Note: We set `random_state` to 0 for reproducibility.

# Gradient Boosting

Gradient Boosting combines weak predictive models (in our case decision trees) in an iterative manner.

Each model corrects the errors of its predecessor, making it highly effective but sensitive to hyperparameter tuning.

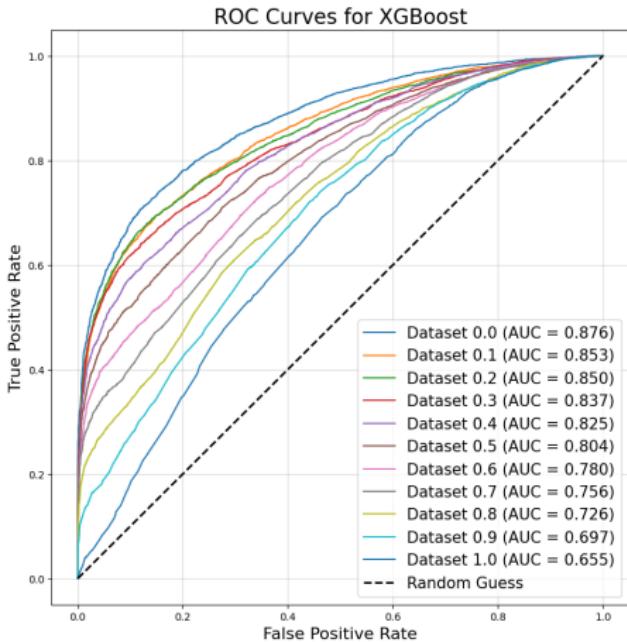
Hyperparameter	Values
<i>n_estimators</i>	125
<i>learning_rate</i>	0.025
<i>max_depth</i>	3
<i>subsample</i>	0.4



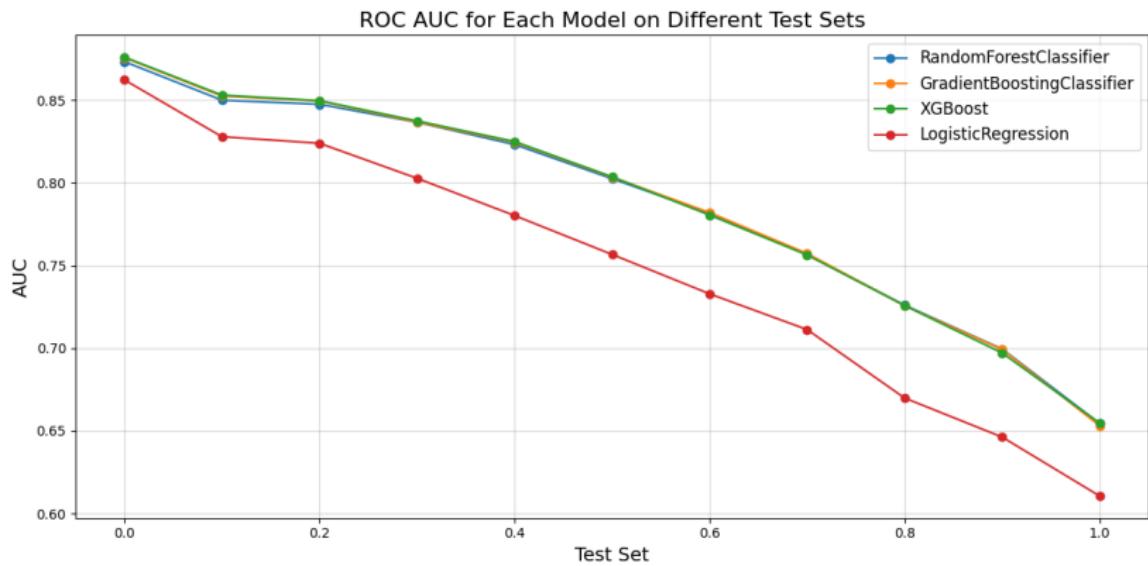
# Extreme Gradient Boosting

XGBoost (*Extreme Gradient Boosting*) is a scalable and efficient gradient boosting framework known for its regularization capabilities and speed.

Hyperparameter	Values
<i>n_estimators</i>	100
<i>learning_rate</i>	0.1
<i>max_depth</i>	6
<i>subsample</i>	0.7
<i>gamma</i>	5



# Performance Comparison



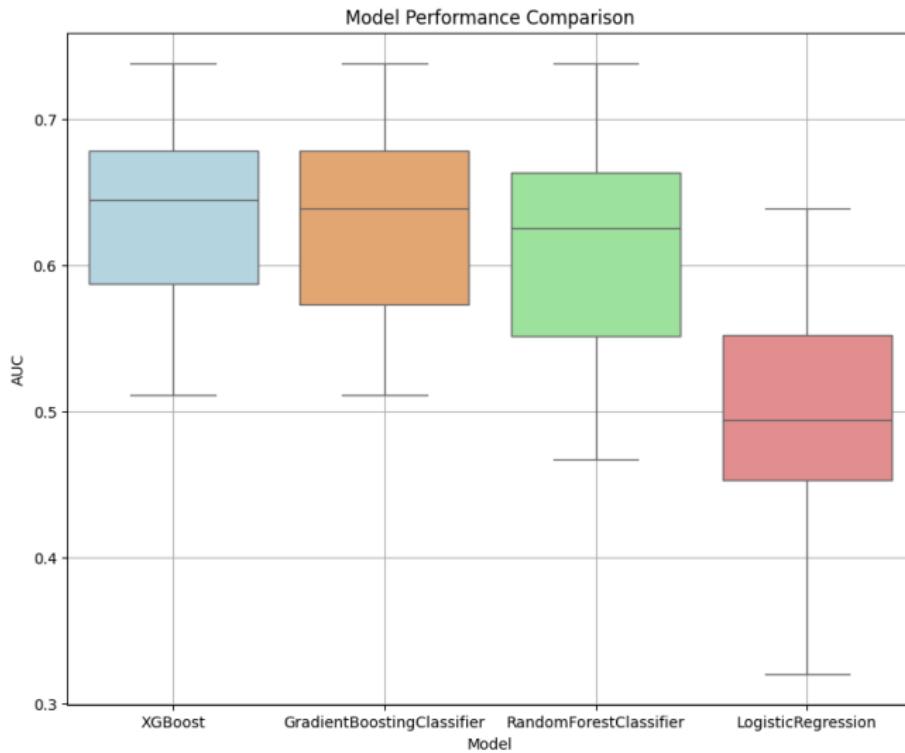
## Statistical Performance Comparison

---

To give statistical support to our study we repeated this experiment  $R = 50$  times. Keeping always the same training set (in order to not train the models several times), for each repetition we:

1. defined a new shifted distribution  $\mathcal{N}(\boldsymbol{\mu}_{\text{shift}}, \boldsymbol{\Sigma}_{\text{shift}})$
2. created 11 **testing** datasets  $\mathcal{D}_\alpha$  with  $\alpha \in \{0.0, 0.1, \dots, 1.0\}$ , where  $\alpha$  represents the mixing probability as before
3. computed the ROC-AUC score for each model on each testing dataset

# Statistical Performance Comparison



## Performance Enhancement

---

## 1. No Prior Shift Knowledge Needed

- Simplifies implementation by eliminating the need for shift estimation.
- Adaptable to various datasets without additional shift information.

## 2. Built-in Regularization

- Prevents overfitting by introducing controlled noise.
- Enhances model generalization on unseen data.

- Random
- Augmentation
- Walk

**Input:**  $Data_{train}$ ,  $Size$ ,  $N$ ,  $\varepsilon$ .

$Data\% \leftarrow$  random subset of  $N\%$  of  $Data_{train}$

**For**  $x_i$  in  $Data\%$

$$x'_i \leftarrow \begin{cases} X_i + \varepsilon & \text{with probability 0.5} \\ X_i - \varepsilon & \text{with probability 0.5} \end{cases}$$

$$y'_i \leftarrow y_i$$

**End For**

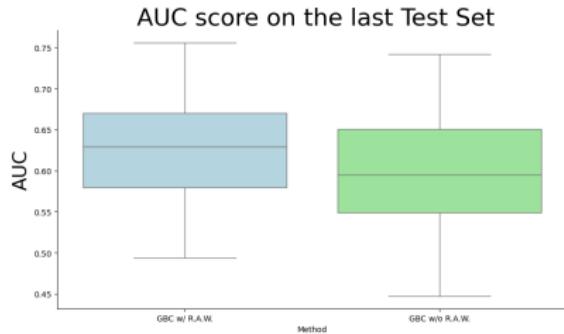
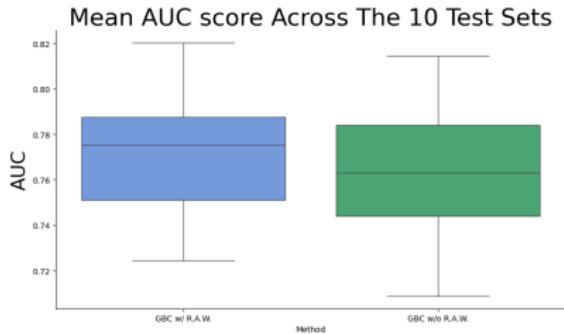
$Data_{aug} \leftarrow Data_{train} \cup Data\%$

$Data_{final} \leftarrow$  Downsample( $Data_{aug}$ ,  $Size$ )

**Return**  $Data_{final}$

# Classify With Gradient Boosting Using R.A.W.

1. Apply the R.A.W. pre-processing method to the training data to address covariate shift.
2. Train a Gradient Boosting Classifier on the augmented dataset.
3. Evaluate the model's performance on shifted test sets.



# A Statistical Analysis Of The Results

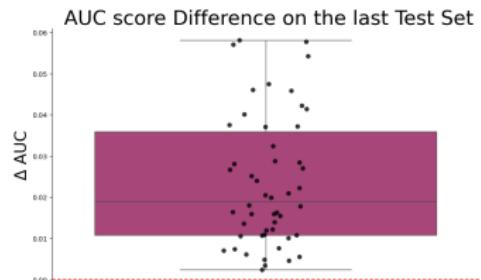
- $H_0$ :

$$\Delta_{AUC} = AUC_{R.A.W.} - AUC_{base} = 0$$

- $H_1$ :

$$\Delta_{AUC} = AUC_{R.A.W.} - AUC_{base} \neq 0$$

- **Test:** Two samples t-test on 50 independent  $\Delta_{AUC}$ .



	$\Delta_{AUC}$	t-stat	p-value	95 % CI
$\Delta_{\overline{AUC}}^*$	0.0083	8.75	$1.39 \times 10^{-11}$	[0.006, 0.010]
$\Delta_{AUC_{last}}^{**}$	0.0235	10.59	$2.86 \times 10^{-14}$	[0.019, 0.028]

\* Mean AUC score difference across all 10 shifted test sets.

\*\* AUC score difference on the most shifted test set.

## Conclusion

---

### How can we address the initial questions posed?

- Covariate shift was explored via synthetic data and repeated simulations.
- Performance degradation highlights the need for "fine-tuned" robust models.
- Every tuned model performed equally across each shift, with the exception of the Logistic Regression.
- **Future work:** Since augmentation showed mitigation potential, we are think about how to further improve the R.A.W. algorithm.

Thank You!