

AMERICAN UNIVERSITY OF BEIRUT

REGULARIZED LOGISTIC REGRESSION  
FOR FAST IMPORTANCE SAMPLING  
BASED YIELD ANALYSIS METHODOLOGY

by

LAMA AHMAD SHAER

A thesis

submitted in partial fulfillment of the requirements  
for the degree of Master of Engineering  
to the Department of Electrical and Computer Engineering  
of the Faculty of Engineering and Architecture  
at the American University of Beirut

Beirut, Lebanon

July 2021

# AMERICAN UNIVERSITY OF BEIRUT

## Regularized Logistic Regression for Fast Importance Sampling Based Yield Analysis Methodology

by  
LAMA AHMAD SHAER

Approved by:

---

Dr. Ali Chehab, Professor  
Electrical and Computer Engineering

Co-Advisor and Committee Chair

*Ali Chehab*

---

Dr. Mohammad Mansour, Professor  
Electrical and Computer Engineering

Member of Committee

**Mohammad M  
Mansour** Digitally signed by  
Mohammad M. Mansour  
Date: 2021.07.10 06:46:18  
-07'00'

---

Dr. Rouwaida Kanj, Associate Professor  
Electrical and Computer Engineering

Advisor

**/Rouwaida  
Kanj/** Digitally signed by /Rouwaida Kanj/  
DN: cn=Rouwaida Kanj, o=American  
University of Beirut, ou=ECE Dept,  
email=110@aub.edu.lb, c=LB  
Date: 2021.07.09 13:59:39 +03'00'

---

Dr. Yasser Mahanna, Professor  
Electrical and Computer Engineering

Member of Committee

*Yasser Mahanna*

---

Dr. Rani Abou Ghaida, Software Engineer  
Xilinx

Member of Committee

*Rani Ghaida*

---

Date of thesis defense: July 8, 2021

# AMERICAN UNIVERSITY OF BEIRUT

## THESIS, DISSERTATION, PROJECT RELEASE FORM

Student Name: Shaer Lama Ahmad  
Last First Middle

Master's Thesis       Master's Project       Doctoral Dissertation

I authorize the American University of Beirut to: (a) reproduce hard or electronic copies of my thesis, dissertation, or project; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes.

I authorize the American University of Beirut, to: (a) reproduce hard or electronic copies of it; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes after: **One \_\_\_ year from the date of submission of my thesis, dissertation or project.**  
**Two \_\_\_ years from the date of submission of my thesis , dissertation or project.**  
**Three \_\_\_ years from the date of submission of my thesis , dissertation or project.**

*Lama Shaer*

Signature

17/9/2021

Date

This form is signed when submitting the thesis, dissertation, or project to the University Libraries

# Acknowledgements

I praise God, the Almighty, for giving me the strength to complete my PhD. I would like to thank Him for the beautiful experiences during this journey that has helped me grow on the personal and professional level. I would like to thank Him for putting the right people in my way to help along the journey.

My sincere gratitude to my PhD advisor, mentor, and role model Dr. Rouwaida Kanj. Thank you for growing my vision and courage all at once by encouraging me and believing in my potentials. Thank you for the beautiful opportunities that you put in my path and your inspirational patience. The lifetime values earned throughout this journey are certainly going to be with me always. I owe every achievement attained in this journey to you! I would like to thank Dr. Ali Chehab for pushing me forward every second since day one. Thank you for being so patient and always listening. I am indebted to you. Thank you for supporting me in my endeavors and always pushing me to give my best.

I would like to thank my mom for doing the impossible to make all this possible. Thank you for making all my achievements possible and for supporting me in unimaginable ways. I would like to thank my dad for all the sacrifices that you do for us. Thank you for all the efforts that you put to provide us with the best always. I am very lucky to have supporting parents like you who gave me

so much strength and patience during this journey. I would like to thank my brother Imad for always being the shoulder to lean on and for supporting me and of course his invaluable advice. I love you all so much.

I want to thank my beloved fiance Rami for his support and patience. Thank you for encouraging me and thank you for your exemplary faith that has in turn grown mine. Thank you for being my guide and mentor in your own ways. May you always be my safe haven even in weakest times.

I want to thank Dr.Ghada El Hajj Fleihan at the SHARP program for encouraging interdisciplinary collaboration at AUB. It was a wonderful and insightful experience for me. I particularly enjoyed Friday's pizza and advise that has certainly aided me through the program and in later experiences as well.

I want to thank my colleagues at BMW for teaching me a lot about large corporations and helping me invest my skills in industrial settings. I enjoyed everyday of this internship and certainly learned a lot. I especially thank my mentors who believed in me and supported me during my time there. I would like to thank my Lebanese friends in Germany who facilitated my internship experience and supported me in every way possible.

I want to thank my friends and colleagues at AUB for making the journey worthwhile. Thank you for the laughs and advice. I will miss this place and cherish all the memories we had together.

I want to thank the Department of Electrical and Computer Engineering for encouraging female representation in the field and for supporting my conference travels. Thank you for the financial and moral aid that has significantly helped me pave my way in my doctoral studies.

# An Abstract of the Thesis of

Lama Ahmad Shaer for Doctor of Philosophy  
Major: Electrical and Computer Engineering

Title: Regularized Logistic Regression for Fast Importance Sampling  
Based Yield Analysis Methodology

With the continuous scaling of semiconductor technologies, new challenges are inflicted on the circuit design yield and reliability. With millions of devices on chip, very tight fail probability requirements are necessary to guarantee reliable designs, and traditional statistical estimation methodologies such as standard Monte Carlo are no longer practical for the statistical estimation of such rare fail events. This is especially true for memory designs where few SRAM cell fails lead to the overall chip fail. Hence, it is necessary to have fast statistical methods to be able to efficiently predict rare fail probabilities. Fast variance reduction methods, such as importance sampling, have been proposed to speed up the statistical simulations. In this work, we propose a regularized logistic regression based fast statistical sampling methodology to speed up the importance sampling methodology flow. We explore several approaches including a proposed greedy heuristic based approach and an improved iterative reweighted least squares-based method for regularized logistic regression. We thus propose a modified Group LARS ap-

proach as an efficient methodology that tracks Newton’s step direction solution evolution from one iteration of the solution to another to efficiently solve for the underlying L1-constrained iterative least squares problem. Data balancing is proven to be very critical and must be employed alongside with regularization to ensure classifier generalization capability and proper separation between classes. We employ data handling techniques in the context of a logistic regression based importance sampling methodology for accurate statistical modeling of rare fail events in memory designs. Synthetic Minority Oversampling Technique (SMOTE) methodology is shown to outperform other data handling methods for purposes of memory yield analysis. Support Vector Machine (SVM) is a popular supervised machine learning methodology that is well known for its efficiency, prediction accuracy and robustness. Typically classifiers tend to treat data points equally, however in certain scenarios this may not be optimal and applying certain weights to the data points can improve the accuracy of the model. We extend what we learn to propose a Best Balance Ratio Ordered Feature Selection methodology for enhanced classifier accuracy for memory design yield modeling. It mimics  $L_0$ -norm regularization through a dual optimization framework and compare it to other implicit data balancing approaches. We critically evaluate the proposed methodologies and prove the efficiency of the proposed approaches by analyzing state-of-the-art FinFET SRAM designs. We demonstrate excellent classifier accuracy and data balancing along with regularization are proven to result in high-fidelity models with excellent yield modeling capabilities.

# Contents

<b>Acknowledgements</b>	<b>v</b>
<b>Abstract</b>	<b>vii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Rare Fail Events: A Major Bottleneck . . . . .	1
1.2 Thesis Statement: Fast Statistical Analysis of SRAM . . . . .	2
1.3 Contributions and Novelty . . . . .	3
1.4 Overview of Results . . . . .	6
1.5 Organization of Thesis . . . . .	7
<b>2 Monte Carlo and Variance Reduction</b>	<b>9</b>
2.1 Motivation . . . . .	9
2.2 Standard Monte Carlo . . . . .	11
2.2.1 Monte Carlo and Variance Reduction Techniques . . . . .	13
2.2.2 Yield Estimation of SRAM . . . . .	15
2.3 Importance Sampling . . . . .	17
<b>3 Feature Selection Methodologies</b>	<b>20</b>
3.1 Motivation . . . . .	20
3.2 Filter . . . . .	21

3.3	Wrapper . . . . .	25
3.4	Embedded . . . . .	27
<b>4</b>	<b>Regularized Logistic Regression for Yield Analysis</b>	<b>29</b>
4.1	Motivation . . . . .	30
4.2	Logistic Regression Overview . . . . .	30
4.2.1	Linear Regression . . . . .	30
4.2.2	Likelihood and Maximum Likelihood Estimation . . . . .	33
4.2.3	Logistic Regression . . . . .	33
4.2.4	Logistic Regression Alternate Formulation and Model Building . . . . .	36
4.3	Overfitting and Regularization Regression . . . . .	39
4.4	Proposed Regularization Framework for Fast Importance Sampling	40
4.5	Proposed Regularization Framework-Heuristics Based Approach .	41
4.6	Cross Validation . . . . .	46
4.7	AIC BIC and Corrected AIC . . . . .	47
<b>5</b>	<b>Group LARS Iterative Reweighted Least Squares Methodology</b>	<b>51</b>
5.1	Least Angle Regression (LARS) Overview . . . . .	52
5.2	Revisiting IRLS-LARS . . . . .	53
5.3	Proposed Group-LARS based IRLS . . . . .	54
<b>6</b>	<b>Data Imbalance Handling Approaches for Accurate Statistical Modeling</b>	<b>60</b>
6.1	Motivation . . . . .	61
6.2	Data Balancing Techniques: Explicit . . . . .	62
6.2.1	OverSampling Methods . . . . .	62

6.2.2	UnderSampling Methods . . . . .	63
6.3	Proposed Data Balancing Logistic Regression Based Method . . . . .	63
<b>7</b>	<b>Best Balance Ratio Ordered Feature Selection Methodology for SVM applications</b>	<b>68</b>
7.1	Motivation . . . . .	69
7.2	SVM Overview . . . . .	74
7.3	Data Balancing Techniques: Implicit . . . . .	78
7.4	2D WSVM . . . . .	80
7.5	BBR-OFS Framework for Yield Estimation Methodology . . . . .	81
7.5.1	Ordered Feature Selection . . . . .	82
<b>8</b>	<b>Results</b>	<b>93</b>
8.1	Experimental Setup . . . . .	93
8.1.1	FinFET DEVICES . . . . .	93
8.1.2	FinFET SOI SRAM . . . . .	94
8.2	Write-Assist Circuitry for FinFET SRAM Design . . . . .	98
8.2.1	Experiments . . . . .	100
8.3	Ordered Feature Selection Methodology . . . . .	102
8.3.1	Phase1: Model Building . . . . .	102
8.3.2	Phase 2: Importance Samples Prediction . . . . .	104
8.3.3	Yield Estimation . . . . .	106
8.4	Data Imbalance Handling Approaches . . . . .	108
8.4.1	Definitions . . . . .	108
8.4.2	Impact of Model Error on Yield Estimation: Imbalanced vs. Balanced Data Sets . . . . .	112
8.5	GroupLARS-IRLS . . . . .	114

8.5.1	Experimental Setup and Yield Analysis . . . . .	114
8.5.2	Performance Evaluation . . . . .	115
8.6	Best Balance Ratio Ordered Feature Selection . . . . .	118
8.6.1	Performance Evaluation: Robustness and Compactness . .	118
8.6.2	Performance Evaluation: BBR-OFS AUC and ROC . . . .	122
8.6.3	Implicit versus Explicit data Balance Ratios: '2D WSVM'	125
8.6.4	Yield Estimation and Runtime . . . . .	127
<b>9</b>	<b>Discussion and Future Work</b>	<b>131</b>
	<b>Bibliography</b>	<b>132</b>

# List of Figures

1.1	SRAM writability simulations. Subject to random process variations, the SRAM cell may fail to write [1]. . . . .	3
2.1	(a) The natural distribution provides a small number of sample points in the region of Fail. (b) The importance sampling distribution $pdf_{IS}(x)$ provides more sample points in the Region of Failure [2]. . . . .	19
4.1	: A simplified illustration of (a) overfitting: model passes through all training points requiring a large number of parameters parameters, versus (b) underfitting: a single parameter (the mean) is used to build the model[3] . . . . .	40
4.2	Methodology Flow Diagram [1] . . . . .	42
4.3	Basic feature selection approach. . . . .	43
4.4	Proposed Heuristic Approach. . . . .	43
4.5	Example of 4-fold cross validation[4] . . . . .	47
5.1	Methodology flow diagram [5]. . . . .	57
6.1	Synthetic sample point generation (golden triangle) or a specific minority sample point (bold line) using SMOTE [6]. . . . .	63

6.2	Methodology Flow Diagram [6]. . . . .	66
6.3	Fail Region (shaded). The ratio of the non-shaded to shaded region reflects the ratio of the number of majority sample points (pass points) to minority sample points (fail points) and was used to compute the theoretical ratio in Figure 6.4 [6]. . . . .	66
6.4	The ratio of minority ‘failed’ sample points to the total number of sample points, obtained in stage 1, can drop to few percent for designs with yield values in the typical ranges of interest. The utilized SRAM cell cross-section has a plurality of random variables [6]. . . . .	67
7.1	2-D illustration of an SVM separating boundary. Support vectors are sample points closest to the boundary [7]. . . . .	76
7.2	Methodology Flow Diagram for the proposed framework. . . . .	82
7.3	K-fold cross Validation [4]. . . . .	86
8.1	Schematic showing a 6-T SRAM . . . . .	95
8.2	Sketch of the SRAM cell cross-section including wordline, bitline and write drivers [8] . . . . .	99
8.3	Capacitive Coupling between Gate and Source boosts the source Voltage(V <sub>ddv</sub> ) when Boost switches ”High” [8]. . . . .	99
8.4	Negative Boost Write Assist is applied to the bitlines during write operation. Cell, write driver, and wordline receive boosted supply voltage [9]. . . . .	100
8.5	Snapshot of the 14nm FinFET SOI technology Die [9]. . . . .	102
8.6	Average number of false predictions versus number of Critical Features for the test set [1] . . . . .	103

8.7	Number of critical features for minimum cross-validation error, and corresponding test set error. Experiments 1 to 6 represent negative bitline boost write-assist based circuit results over [0.35V-0.4V], and experiments 7 to 10 represent the standard circuit experiments over [0.4-0.43V] [1]. . . . .	104
8.8	Standard Design phase 2 pass/fail prediction for the importance sample false negatives(false pass) were reported for this set of experiments [1] . . . . .	105
8.9	Write Assisted Boosted design phase 2pass/fail prediction for the importance sample points [1]. . . . .	105
8.10	normalized for max and min fail probabilities. True: represents circuit simulation based approach. Predict: represents proposed methodology [1] . . . . .	107
8.11	Probability Convergence comparison. Results are normalized for max and min fail probabilities.True: represents circuit simulation based approach. Predict: represents proposed methodology [1] . . . . .	107
8.12	Yield Prediction for Standarddesign. Results are normalized for max and min yield. True: represents circuit simulation based approach. Predict: represents proposed methodology [1] . . . . .	108
8.13	Confusion Matrix [6] . . . . .	109
8.14	Data sample points projected to a 2-D quadratic feature space for (a) the unbalanced original data set and (b) set oversampling via SMOTE [6] . . . . .	110
8.15	Recall rate for the different data balancing techniques [6]. . . . .	110
8.16	Precision rates for the different data balancing techniques. . . . .	111
8.17	Cell probability for thr different data balancing methods [6]. . . . .	112

8.18	Yield difference between the circuit simulation based approach and the yield estimated using logistic regression model based approach for the different balanced/imbalanced data sets. We clearly notice a conservative yield estimate when SMOTE is adopted [6]. . . . .	113
8.19	Runtime savings for IRLS-GroupLARS compared to IRLS-LARS for an example Experiment. Embedded table summarizes results for the different experiments [5]. . . . .	116
8.20	$CV E_{min}$ and the critical number of features [5]. . . . .	117
8.21	Prediction accuracy for IRLS-GroupLARS. . . . .	117
8.22	(a) Probability Convergence results for $IRLS_{GroupLARS}$ estimate $\hat{P}_f$ . Golden ( $P_f$ ) represents circuit simulation based approach. b) Yield estimate [5] . . . . .	118
8.23	Cross Validation error versus $\lambda$ for an example $L1$ -SVM path [7]. .	120
8.24	$CV E_{min}$ for the different experiments for the fully balanced and raw ratios for the uniform sampling datasets [7]. . . . .	121
8.25	Variable reduction for the proposed BBR-OFS models versus is higher than that of $L1$ -SVM $_{\rho}$ , i.e., $L1$ -SVM on raw data and $L1$ -SVM $_{\beta}$ , i.e., $L1$ -SVM on the different balanced datasets. . . . .	121
8.26	ROC curves for an example raw data set indicate improved AUC value for the OFS $_{\rho}$ solution compared to $L1$ -SVM. AUC values for the ordered feature selection iterations are also presented [7]. . . .	124
8.27	The AUC value for the BBR-OFS solution is among the best compared to the raw data set solutions and the other solutions employing the same data balancing ratio with varying feature sets [7]. . . . .	125

8.28	2DWSVM $\pi$ -Adjusted ratio for the best solution in 2D solution surface space [7]. . . . .	126
8.29	Yield error, $ \sigma_{err}  =  \sigma_{hat} - \sigma_{ref} $ , where $\sigma_{ref}$ is the yield for the pure-circuit simulation based approach. $\sigma_{hat}$ is the yield corresponding to the model based importance sample points evaluation for the different approaches [7]. . . . .	128
8.30	Scatter yield plot for pure circuit simulation based approach (golden), versus BBR-OFS (proposed). BBR-OFS demonstrates an average yield sigma error of 0.19 sigma [7]. . . . .	129

# List of Tables

2.1	The Number of Monte Carlo Sample points versus $P_f$ for the ratio $\frac{\sigma_{P_f}}{P_f} = 10\%$ [2]. . . . .	17
4.1	Pseudo Code for batch gradient descent [2]. . . . .	32
8.1	Summary of the designs used. (a) The Standard design refers to the selective boost design with no write assist. (b) The Boosted design refers to the presence of negative bitline boost for write assist.	101
8.2	The number of False Negatives decreased for most methods (except NearMiss) compared to the imbalanced data set for stage 2 data prediction. NearMiss systematically got worse suffered as stage 1 data imbalance increased. . . . .	111
8.3	Relative Yield Error (for appreciable yield 10%) $\text{Abs}((Yield_{modelBased} - Yield_{CktSim}) / Yield_{CktSim}) * 100$ . . . . .	113
8.4	Phase II Speedup and Runtime in seconds. . . . .	130

# Chapter 1

## Introduction

### 1.1 Rare Fail Events: A Major Bottleneck

While aggressive technology scaling has been a major driver behind the semiconductor industry, it has exacerbated the process, voltage and temperature variations for nano-scale technologies. These variations manifest themselves as unavoidable uncertainties in circuit performance and reliability [10]. Due to their density requirements, memory designs use the smallest devices on the chip. Hence, they rely on the most aggressive design rules and tend to suffer from variability most. For SRAM designs this leads to unavoidable manufacturing variations between neighbouring transistors of the same cell. With millions of cells in an array, designers enforce tight constraints on cell fail probabilities with requirements of less than one-part-per-million fails. This is particularly true for memory designs where a few SRAM cells failing can lead to a system failure [11]. Capturing these fail probabilities with high-fidelity is an indispensable part of the design cycle. Monte Carlo analysis is traditionally used as a statistical means to estimate design failure probabilities. However, Monte Carlo simulation method

is not practical for estimating such rare fail probabilities, and it would require millions of SPICE simulations to capture a single failing point in the event of designs targeting rare fail events [12]. Therefore, efficient statistical models should be developed to capture these rare fail events.

## **1.2 Thesis Statement: Fast Statistical Analysis of SRAM**

It is therefore critical to develop fast and reliable statistical methods for the purposes of rare event estimation. Several methodologies have been proposed to speed up memory rare event estimation. In [13], the authors developed statistical models for the SRAM DC noise margin during the functional read and write. The authors in [14] built models for the probability distributions of the SRAM cell Read and Write access times. The authors in [12] proposed Statistical Blockade as a fast statistical methodology that filters tail samples to build models for the tail distributions and predict fail probabilities. They rely on margining and true circuit simulations to eliminate tail false positives. Circuit-simulation based Importance Sampling methods [15, 16, 17, 18] are popular variance reduction techniques that were developed to efficiently accelerate Monte Carlo simulations for rare fail estimation of SRAM cell dynamic margins. By biasing the sampling towards the critical fail region, Importance Sampling methods offer several orders of magnitude speedup over traditional Monte Carlo [15]. Accurately estimating rare memory fail probabilities is a challenging task that is closely related to machine learning. Recently, machine learning techniques garnered major interest in the area of memory design to develop accurate and fast yield models for memory design yield enhancement [19]. Often functional fails translate into

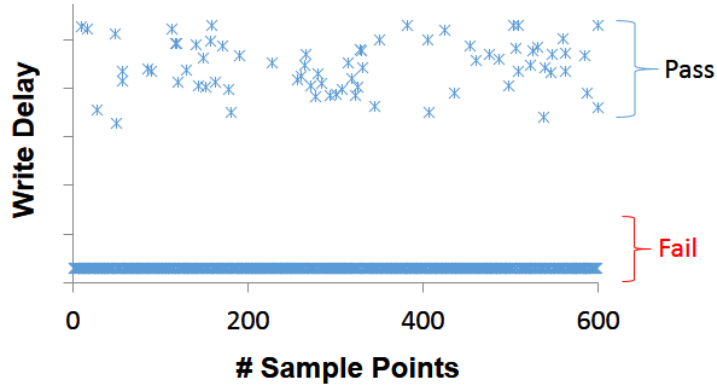


Figure 1.1: SRAM writability simulations. Subject to random process variations, the SRAM cell may fail to write [1].

discontinuities in the design metric space. Figure 1 illustrates SRAM cell writability sample points. While it is possible to record the delay of a successful write, there is no change in the cell content when the cell functionally fails to write. From a dynamic margin perspective, we simply return 0 for a fail to write. Traditional response surface modeling approaches do not capture this response, and binary classifiers such as the logistic regression must be employed. Indeed, machine learning has become a vital tool for yield modeling of memory designs in the presence of process variation effects. This area, nevertheless, faces several challenges due to the rare fail event nature of the array yield estimation problem [20]. A methodology that accurately models SRAM cell behavior and efficiently captures the rare fail events is necessary to tackle this problem.

### 1.3 Contributions and Novelty

This dissertation presents novel methodologies that allow for fast statistical analysis of memory design. The machine learning based methodologies enable fast analysis of memory designs, provide highly accurate yield estimation, and im-

proves analysis runtime. Below is a list of contributions of the thesis.

1. An ordered feature selection based logistic regression methodology for a fast statistical analysis framework using importance sampling methodology helps avoid overfitting and enables regularization for accurate yield modeling. We propose ordered feature selection along with cross-validation as an approximation for the L0-norm regularization. The methodology is demonstrated on state-of-the-art FinFET SRAM design with write boost circuitry. Writeability analysis results demonstrate high accuracy both at the circuit simulation and yield estimation levels. We observe less than 3% sigma yield error compared to pure circuit simulation approaches. All this comes at significant savings in runtime.
2. Data handling techniques in the context of a logistic regression based importance sampling methodology. By relying on the SMOTE methodology, we are able to improve the model recall and precision and reduce the data set induced error on the final yield estimate down to a relative error of 5% compared to 18% for the imbalanced data set-based approaches.
3. We propose a scalable and efficient L1 regularized logistic regression methodology with application to importance sampling based rare event estimation. It relies on a quadratic approximation to the non-linear optimization problem. At the core, lies a Group LARS-based methodology that tracks Newton's step direction solution evolution from one round to another, and weighted directions to efficiently solve for the underlying L1 constrained linearized problem. When applied to the yield analysis of a 14nm FinFET SRAM design, our results show upto 14x speedup for Group LARS iterations compared to pure LARS based approach, and we report 98.7

% accuracy and  $0.09\sigma$  average error compared to pure circuit-simulations approach.

4. A data balancing scheme with varying balance ratio to handle the class imbalance problem due to the typically low proportion of minority fails in memory design problems. This allows for identifying the Best Balance Ratio (BBR) among a set of ratios ranging between the natural and fully balanced dataset distribution for enhanced classifier generalization capabilities, thereby not ruling out the naturally occurring class distribution in accordance with literature.
5. Extending Ordered Feature Selection (OFS) approach [1] in the context of SVM as an efficient classifier. We reformulate it as an approximation to the otherwise non-convex SVM  $L_0$ -norm regularization that aims to identify the relevant SVM features for the respective imbalanced datasets with the different adjusted balance ratios. For this, we build upon a semi-smooth Newton coordinate descent  $L_1$ -SVM solution engine [21] to iteratively identify the relevant features for a given dataset. While the  $L_1$ -SVM is often performed as a relaxation to the  $L_0$ -norm in terms of augmenting the optimization constraints, we demonstrate enhanced accuracy for our  $L_0$ -norm approximation.
6. We study an implicit 2-dimensional surface solution methodology (2DWSVM) targeting imbalanced datasets classification [22] that employs adjusting the classifier learning algorithms to better recognize the minority class. The methodology identifies the optimal cost function weights for the SVM classes as determined among a derived family of solutions. We rely on this approach as our reference implicit data balancing methodology and criti-

cally evaluate its classification and balancing capabilities compared to the proposed explicit approach.

7. We study the yield of state of the art industrial 14nm FinFET SRAM design. We compare our proposed BBR-OFS methodology to the  $L_1$ -SVM approach, OFS on the natural distribution and the implicit 2DWSVM approach. Our results demonstrate that BBR-OFS outperformed other methods in terms of accuracy while maintaining significant improvement in terms of the runtime efficiency compared to pure circuit simulation based approach. This is manifested both at the model prediction and yield estimation levels.

## 1.4 Overview of Results

The proposed methodologies were tested on 14nm state-of-the-art FinFet SRAM and have demonstrated high efficiency and significant speedup compared to pure-circuit simulations based approaches.

With respect to the ordered feature selection based logistic regression methodology, the framework was computationally efficient and resulted in high-fidelity models. We observe around 4.5% false prediction rate for the importance sample points prediction. This signifies accurate yield prediction for the rare fail events. As well, the proposed framework achieves significant savings in runtime.

As for the Group LARS based iterative re-weighted least squares methodology for efficient and accurate regularized logistic regression, the approach benefits from GroupLARS inherent weighted least angle direction to pass the critical feature group from one solution round to another. It enables a scalable and efficient regularization framework that speeds up the search for the best model. We

demonstrate high accuracy for the resulting classifier for the importance sample points both in terms of the number of false predicts as well as the yield estimate. For the experiments understudy, we demonstrate up to 14x speed up when the model is developed using the maximum number of desired variables compared to the iterative LARS based approach. This is expected to increase even more as the dimensionality of the problem increases.

The Data balancing based framework was among the major game changers as it proved the importance of the necessity of balancing datasets. We demonstrate that the synthetic minority oversampling technique outperforms other methods. We report more than 70% reduction in the number of False Negatives compared to natural imbalanced data set. We also report in terms of yield estimate on average a small relative error rate in the yield estimate for the balanced data of 5% compared to the pure circuit simulation based approach. Whereas, an average of 18% relative error rate obtained for the imbalanced data set-based approaches. Last but not least, our proposed BBR-OFS SVM methodology has proven to improve yield modeling compared to other SVM approaches. The results demonstrate that BBR-OFS outperforms other methods and offers an efficient estimate of the memory reliability metric with an average error of 0.19 sigma for the yield estimate and runtime improvement of 179x compared to the pure-circuit simulation based approach.

## 1.5 Organization of Thesis

The dissertation is organized as follows. Chapter 2 gives an overview on Monte Carlo and Importance Sampling. Chapter 3 provides background review on Feature Selection Methodologies. Chapter 4 provides an overview of the logistic

regression based method that employs feature selection. Chapter 5 presents data balancing techniques. It details the data balancing method employed in the framework discussed in chapter 3 for rare fail event estimation. Chapter 6 presents Group LARS based methodology. Chapter 7 presents Support Vector Machine and Data Imbalance, highlighting the effect of data imbalance on SVM and solutions proposed. It presents the novel best balance ratio ordered feature selection technique employed for prediction of rare fails. Chapter 8 discusses our implementation and experimental results. Finally, Chapter 9 concludes this work and proposes future research related to statistical analysis of memory designs.

# Chapter 2

## Monte Carlo and Variance Reduction

The objective of this chapter is to give a general overview about Monte Carlo method and variance reduction techniques for fast statistical analysis of memory designs in the event of rare fails. We start by giving an overview about Monte Carlo.

### 2.1 Motivation

Aggressive technology scaling approaching the sub 20-nm threshold throughout the decades compromised the reliability of integrated circuits due to the resultant statistical variations. These variations may be attributed to random dopant fluctuations, oxide thickness variation, line edge roughness and short channel effects among other effects. As such, circuit designers need to handle and analyze the underlying statistical nature of these variations. Traditionally, circuit designers relied on process corner analysis to achieve this, however this approach was infea-

sible due to the large amount of variations that arise. Robust analytical solutions to improve circuit performance and parametric yield have been introduced[23]. Nevertheless, in the realm of statistical methodologies, Monte Carlo was the real winner.

The Monte Carlo method reproduces a statistical process to compute an approximation of a deterministic quantity. In circuit design and for variables with a parametric inherent nature, Monte Carlo employs a probability distribution (uniform or normal) to build a model of possible outcomes[24]. Monte Carlo gained popularity in circuit design becoming the gold standard in performing statistical simulation of circuits and for yield estimation during the design phase [25]. In standard Monte Carlo application, the yield estimate depends on the sample size and not the feature space. Alas, Monte Carlo presents several advantages at an expensive cost due to the large number of computations performed. Each Monte Carlo run requires thousands of SPICE simulations. This only motivated members of the EDA community to seek alternative and less costly methods to the Monte Carlo approach. While response surface model approaches did improve that speed yet it was at the cost of accuracy. Accuracy requirement can be as stringent as requiring less than one fail per million. Hence, there is a significant need to develop fast and reliable statistical analysis methodologies for rare event estimation.

Many methodologies have been proposed to speed up memory yield estimation. For example, in [15], the authors introduce probability distribution models for the SRAM DC noise margin while performing the read and write operations. Dynamic margins proved to be reliable resource for circuit designers that allowed them to confidently evaluate SRAM fails [18]. Thus, circuit-simulation based Importance Sampling methods [15, 16, 17, 18], are commonly used variance re-

duction techniques that were introduced for fast statistical analysis of SRAM cell dynamic margins. Importance sampling mainly focuses on regions of importance and adjusts the estimate to account for oversampling from these areas [11].

In the rest of the chapter, we will review the standard Monte Carlo method and variance reduction techniques that were later proposed. We also present the Importance Sampling method for Monte Carlo estimation which is basis to our main contributions in this thesis.

## 2.2 Standard Monte Carlo

Early Monte Carlo experiment was conducted by Buffon in the needle experiment which took place in the 1700s. Monte Carlo was used to estimate pi [26]. Ulam later utilized the Monte Carlo approach to analyse the scattering of the neutrons [27]. Monte Carlo gained widespread attention and became an famous statistical tool in the various fields of sciences and engineering. Monte Carlo methods are used for approximating yet obtaining accurate numerical solutions to mathematical problems such as numerical integration, statistical inference, and reliability estimation.

Monte Carlo is used to estimate an Integral such as:  $\text{Int} = \int_a^b f(x)dx$ . Monte Carlo method is the preferred method for integration in high dimensions. It uses random sampling by sampling  $N$  points  $X_1, X_2, \dots, X_n$  according to a certain density function  $p$ , and then computing the estimate using:

$$Y_N = \frac{1}{N} \sum_{i=1}^n \frac{f(X_i)}{p(X_i)} \quad (2.1)$$

The result of the above ,  $Y_N$ , is a random variable and depends on the samples  $N$ . This is referred to as the the Horvitz-Thompson estimator [28].

If the samples are chosen randomly and uniformly, then the equation used to compute  $Y_N$  reduces to:

$$Y_N = \frac{1}{N} \sum_{i=1}^n f(X_i) \quad (2.2)$$

This has a similar form to the quadrature rule but the samples are random.[29]

Monte Carlo integration is particularly useful for the field of graphic design where there is a dire need to perform integrals in high dimensional space. It is also simple and general. Thus, this method is very appealing to use. Monte Carlo is also a popular tool for statistical inference. For example, it can be used to find the inference of a mean  $\mu_x = \int x pdf(x) dx$  of the variable  $x$  with a probability density function  $pdf(x)$ . When using Monte Carlo, it is important to be able to generate random numbers easily from the distribution with the given density [30].

In the context of Reliability Estimation, Monte Carlo is used for estimating a probability of fail  $P_f = \int_R pdf(x) dx$  as determined by a region of fail  $R$ . This is of main interest as it is directly relevant to the work throughout the thesis. Monte Carlo has been very prevalent in the field of integrated circuits for verifying high yield in the event of random process variations.

Although Monte Carlo offers several advantages such as being simple to use and capable of dealing with high dimensional integrals, it does come at an expensive cost. The major drawback of Monte Carlo techniques is the slow convergence. There are several variance reduction techniques which we will talk about later, but despite that, these methods converge slowly. Hence, they are only used when no other viable option presents itself [31].

### 2.2.1 Monte Carlo and Variance Reduction Techniques

Monte Carlo Statistical method relies on random sampling to be able to infer information about the general population [32]. When it comes to memory designs, the statistical inference method become particularly appealing for proportions. A memory cell can be seen as a Bernoulli trial with a predetermined probability of fail,  $P_f$  and the memory array size depicts the number of trials,  $n$ . The designer relies on the estimate of  $P_f$  to approximate the overall chip yield using the equation:

$$Yield = (1 - P_f)^n \quad (2.3)$$

Standard Monte Carlo arises as a traditional statistical methodology for estimating the probability of fail. Given the tight constraint of less than one part per million fail, circuit designers must seek reliable methods to ensure very low cell fail probability. This makes Monte Carlo very slow, and necessitates the use of variance reduction techniques.

The accuracy of Monte Carlo is affected when utilized for rare fail estimation. Hence, variance reduction techniques enhance the accuracy of the estimate for a given number of sample points compared to standard Monte Carlo. Most common variance reduction methods include [33] :

1. Control Variates method: Assuming that the standard Monte Carlo method aims to estimate the expected value of a variable  $Y$ , according to

$$\mu_y = \frac{1}{N} \sum_{i=1}^N Y_i \quad (2.4)$$

The control variates method introduces a variable  $Z$  correlated to  $Y$ . It is

based on the condition that when the variance of  $Y$  is large, the variance of  $(Y - cZ)$  is small. The parameter  $c$  is ratio of the covariance of  $Y$  and  $Z$  to the variance of  $Z$ .

$$c = \frac{Cov(Y, Z)}{Var(z)} \quad (2.5)$$

As such, this method approximates the expected value or the mean of the variable  $Y - cZ$ ,  $\mu_{CV}$  as an unbiased estimate of  $\mu_y$ .

$$\mu_{CV} = \frac{1}{N} \sum_{i=1}^N (Y_i - c(Z_i - \mu_Z)) \quad (2.6)$$

This method is a good way that can be used to reduce error and variance by replacing approximate operations with exact ones.

2. Antithetic Variables is another variance reduction method that is quite the opposite of the control variates method. For every sample path, this method also samples the antithetic path, i.e., for every direction, it samples the opposite direction. Rather than incrementing the samples by using a positively correlated function, the estimator is merged with a negatively correlated one. This leads to a reduction in the variance and lowers the sample points to be generated.

An example of this method would be a Monte Carlo integration of the function  $f(z) = z$  in the interval  $[0, 1]$  by evaluating the integrand at the points  $z_i$  and  $1 - z_i$ .

3. The technique of the stratified sampling variance reduction works on stratifying the . The variable space  $\Omega$  in the stratified sampling method is

divided into several disjoint subspaces  $\Omega_l$  ( $l=1,2,\dots,L$ ), and  $\Omega_l$  satisfies the following relationships:

$$D = \cup_{i=1}^L D_i D_i \cup D_j = \phi(i \neq j, i, j = 1, 2, \dots, L) \quad (2.7)$$

where  $L$  is the subspaces divided, and  $\phi$  is the empty set.

Depending on how much the subspace impacts the fail probability, the corresponding number of samples is given to the subspace.

Assuming that  $P_l$  ( $l=1,2,\dots,L$ ) is the probability of  $\Omega_l$ , and  $l \Omega_F$  is the failure probability of  $\Omega_l$ , then  $P_l$  can be expressed as:

$$P_l = \int_{\Omega} f(x) dx \quad (2.8)$$

The stratified sampling technique is a variance reduction method that has lower computational cost, but requires choosing an appropriate stratified strategy.

4. Importance Sampling: The natural distribution is distorted to place more emphasis on sampling the critical fail regions. The estimate is unbiased by correcting for the true to distorted probability density function ratios.

## 2.2.2 Yield Estimation of SRAM

SRAM arrays use the smallest devices on chip and are majorly impacted by process variations. These variations may be due to random dopant fluctuations, oxide thickness variation, line edge roughness and short channel effects. The design of large arrays requires design targeting for five or more standard deviations. This is especially a challenge for the design of near-minimum-sized SRAM

cells targetting for low-power applications. If unresolved, this can affect also the high-density criteria.

Assuming that a circuit simulation is performed under a set of process variations  $x_p$ , we compute a set of performance metrics denoted as  $f_i(x_p)$ . Also, given the set  $f_i^0$  that represents the performance metric fail criteria, we identify the indicator function  $I(x)$  as follows.

$$I(x) = \begin{cases} 0 & f_i(\vec{x}_p) \leq f_i^0 \\ 1 & \text{o.w.} \end{cases}$$

The probability of fail  $P_f$  can be approximated using

$$P_f = E(x) = \int I(x) pdf(x) dx \quad (2.9)$$

The final result of Monte Carlo's is not deterministic and differs between different random sample sets. The variance of the approximate probability of fail  $P_f$ ,  $\sigma_{P_f}^2$  is defined as follows:

$$\sigma_{P_f}^2 = \frac{(1 - P_f) * P_f}{N} \quad (2.10)$$

Therefore reducing variance requires having a larger number of sample points. In case of small probabilities, the number of sample points becomes larger and ultimately requires a larger number of circuit simulations. (see Table 2.1 for an example). Therefore, the variance can be lowered by increasing the number of sample points. For small probabilities, the number of sample points and hence the number of circuit simulations becomes extremely large (see Table 2.1 for an example). Hence, it is infeasible to rely on traditional Monte Carlo techniques for accurately predicting such rare fails probabilities. Statistical analysis techniques with enhanced robustness such as variance reduction methods are essential to

improve the accuracy of the estimate.

Table 2.1: The Number of Monte Carlo Sample points versus  $P_f$  for the ratio  $\frac{\sigma_{P_f}}{P_f} = 10\%$  [2].

Probability of fail	Number of Monte Carlo points $N$
$10^{-3}$	$10^5$
$10^{-4}$	$10^6$
$10^{-5}$	$10^7$
$10^{-6}$	$10^8$

As such, there are many factors that promulgate the selection of the best yield estimation approach. The candidate approach is limited by the number of simulations that can be provided, and the The number of possible circuit simulations depends on the computing resources available. This number in turn is dependent on the real failure rate and desired confidence level. Monte Carlo becomes less efficient in the episodes of estimating rare fail events. Variance reduction methods arise as enhanced and more reliable tools for estimation given number of sample points compared to standard Monte Carlo. Popular variance reduction methods include [34]: Control Variates method, Antithetic Variables, Stratified Sampling, and Importance Sampling.

## 2.3 Importance Sampling

Importance Sampling (IS) has been proposed based on the insight that drawing samples in the likely-to-fail regions with a distorted sampling distribution can improve the estimation accuracy and meanwhile accelerate the estimation convergence. The choice of distorted distributions has a significant impact on the performance of IS.

In [32], Importance Sampling was proposed to speed up Monte Carlo analysis. The theory is based on equation 2.11, where the fail probability can be estimated using the natural distribution  $p(x)$  or alternatively a distorted importance sampling distribution  $g(x)$ . The methodology relies on two phases. The first phase performs uniform sampling in the space to mark the center of gravity of the fail region and determine the recommended shift for  $g(x)$ . Meanwhile, the second phase creates importance sample points according to the function  $g(x)$ .

$$prob = \int f * p(x) dx = \int (f * \frac{p(x)}{g(x)}) * g(x) dx \quad (2.11)$$

Figure 2.1 presents a case where the uniform distribution is used to generate more sample points in the tail region compared to the natural Gaussian pdf. There exists three main forms of Importance Sampling:

1. Integrated Importance Sampling.
2. Ratioed Importance Sampling.
3. Regression Importance Sampling.

In the case of integrated Importance Sampling, the Monte Carlo approximation in is modified in a way such that it accounts for the distortion of the natural sampling distribution as indicated in 2.12.

$$\widehat{P}_f = \frac{1}{N} \sum_{i=1}^N I(x) (46) \int f(x) pdf(x) dx = \int f(x) \frac{pdf(x)}{pdf_{IS}(x)} dx \quad (2.12)$$

Hence, when sampling from  $pdf_{IS}(x)$ , the importance sampling based proba-

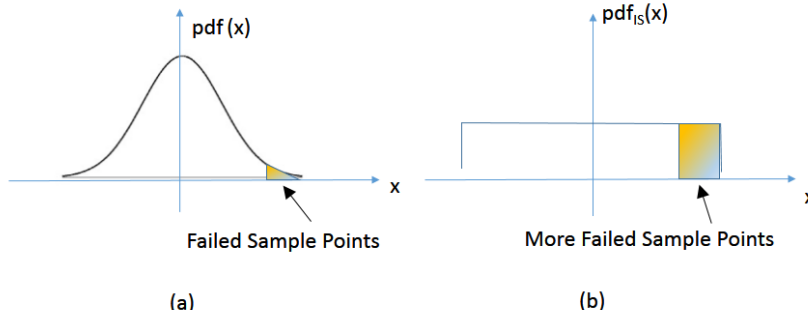


Figure 2.1: (a) The natural distribution provides a small number of sample points in the region of Fail. (b) The importance sampling distribution  $pdf_{IS}(x)$  provides more sample points in the Region of Failure [2] .

bility of fail estimate,  $\hat{P}_{fIS}$ , and its variance,  $\sigma_{P_{f:IS}}^2$   $\sigma_{pf:IS}^2$ .

$$Y(x_i) = I(x_i) * \frac{pdf(x_i)}{pdf_{IS}(x_i)} \quad (2.13)$$

$$\hat{P}_{fIS} = \frac{1}{N} \sum_{i=1}^N Y(x_i) \quad (2.14)$$

$$\sigma_{pf:IS}^2 = \frac{1}{N(N-1)} \sum_{i=1}^n (Y(x_i) - (\bar{Y})) \quad (2.15)$$

# Chapter 3

## Feature Selection Methodologies

The objective of this section is to give a general overview about the main feature selection methodologies that we later on build our main contributions upon.

### 3.1 Motivation

Machine learning problems are present in different domains such as robotics, pattern recognition, and biomedical applications. Given a set of training examples associated with a certain output, the objective of machine learning is to construct a relationship between the input sample points and the outcome. The main goal really is to find the outcome for a new test sample point. This is done by building a model capable of generalizing to predict the outcome using certain parameters. However, before building the model, a proper data representation must be chosen[35].

Data is represented using a set of variables or features and data representation is very much domain-specific. Feature construction is concerned with converting raw data into a feature set and is defined as data preprocessing. The data prepro-

cessing steps includes standardization, normalization, feature extraction, linear and non-linear space embedding, and non-linear expansion. The dimensionality is unaffected when standardization and normalization is applied. However, it can be reduced or increased by space embedding methods (PCA) or non-linear expansions respectively. While adding many features to the model enhances the predictive capability of the model, it comes at the cost of having a higher dimensions. Thus, arises the need for feature extraction. While the main reason for performing feature extraction is to lower the dimensionality, it may have other advantages. These advantages include reducing the data due to the storage limits, saving resources for data collection and enhancing accuracy[36]. There are three main types of feature extraction methods: Wrapper, filter, and embedded which we will discuss next.

## 3.2 Filter

Feature selection encompass algorithms of different types. Filter method based feature extraction focuses on filtering feature that can be less informative than others. It is a computationally efficient method. This technique uses a feature filter that computes relevance of a feature subset to the outcome. This may be computed for individual attributes (by ranking them). A good feature should not be redundant and should not be correlated with other features. Computing the correlation between a feature and a class is critical [37].

Variable ranking as a filter method was proposed by Kohavi and John [38]. Fisher's criterion can be used to rank variables for Fisher's Linear Discriminant classifier[39]. Variable ranking as a filter method is not only used for reducing dimensions but also to avoid overfitting since it gives way to bias and has less

variance [40]. Assuming that the input variable vector is  $\mathbf{x}$  and  $x_i$  is the  $i^{th}$  element in the vector. Furthermore, we denote  $\mathbf{y}$  to be the output vector. Different ranking criterion are used for ranking. These include Mutual Information, Pearson correlation criteria, Chi-square test, Correlation coefficients, etc.

Using correlation coefficients for filter method is the most common way. They are also popular as they do not require discretization of continuous variables.

The contingency table is a popular statistical method to analyze the correlation between two categorical variables. Given a dataset with  $l$  training sample points, it can be split into subsets of  $L_{ij}$  sample points which belong to class  $y_i$ ,  $i = 1..K$  and have a specific feature value  $x_j$ . The rows of the  $L_{ij}$  matrix are summed up to obtain marginal distribution  $L_i$ . of samples over classes is obtained. Meanwhile, the columns distribution  $L_j$  of sample points over the values  $x_j$  is obtained. The measure of how strongly the variables  $x$ ,  $y$  are associated is obtained using:

$$\tilde{\chi}^2 = \sum_{ij} \frac{(L_{ij} - l_{ij})^2}{l_{ij}} \quad \text{where } l_{ij} = \frac{L_i \cdot L_j}{l} \quad (3.1)$$

$l_{ij}$  reflects the average number of outcomes given that  $x$  and  $y$  are independent. If the attribute and the outcome were independent then  $l_{ij} = L_{ij}$ . On the other hand, if the difference was big this would mean that they are dependent[41].

For continuous variables, the Pearson correlation coefficient is another popular metric used for the ranking of variables. Given an input vector  $X$  and output  $Y$ , the correlation coefficient is the absolute cosine value of the vectors  $X$  and  $Y$ , after subtracting the means from both.

$$r = \frac{E(XY) - E(X)E(Y)}{\sqrt{\sigma^2(X)\sigma^2(Y)}} \quad (3.2)$$

If the attribute and target are not correlated  $r = 0$  otherwise it is a positive or a negative number between -1 and 1. The correlation can be negative or positive. If the relation between the input features and the output target is monotonic, then this approach is convenient [42].

For data with binary outcomes  $\{y_+, y_-\}$ , another criterion commonly used is the signal-to-noise ratio computed using the Separation of the means of the class distributions [43]. It is given by the following equation:

$$\mu(X, Y) = \frac{\mu(y_+) - \mu(y_-)}{\sigma(y_+) + \sigma(y_-)} \quad (3.3)$$

The square of this coefficient resembles the ratio of between-class to within-class variances, known as the Fisher criterion [44].

For a group of  $k$  features that has already been selected, the correlation coefficients may be used to estimate correlation between this group and the class which reflects the relevance of this group and the class. Also, the correlations between the features may be computed. Relevance of a group of features grows with the correlation between features and classes. On the other hand, the higher inter-correlation there is between the features the lower the relevance [45].

If the type of data used is ordinal then non-parametric Spearman's rank methods should be used[46].

Relief is another family of algorithms that was proposed for feature weighting. This methods infers the discriminative capability of a feature and how much it can distinguish neighboring sample points. The Relief Relevance index  $J_R(X)$  for feature  $x$ , having two nearest neighbor points  $x_s$  of same class and  $x_d$  of different class, is increased by the value proportional to  $|X(x) - X(x_d)|$  and decreased by a value proportional to  $|X(x) - X(x_s)|$ . Relevance must be larger for features

that are capable of separating vectors from different classes and vice-versa. The ReliefF algorithm was an extension to the Relief algorithm that was introduced to address multiclass problems. The dependence between the class and feature may be computed using the difference between probability distributions[47] given  $K$  classes. Kolmogorov proposed the difference metric:

$$D_K(X, Y) = \sum_i \sum_{j=1}^K |P(y_j, x_i) - P(y_j)P(x_i)| \quad (3.4)$$

This measure resembles a lot the chi-square statistic mentioned previously. Feature relevance can also be computed using information theory indices. Entropy is an information theory measure given below:

$$H(X) = - \sum_{i=1}^K P(y_i) \log_2 P(y_i) \quad (3.5)$$

where  $P(y_i) = \frac{m_i}{m}$  is the fraction of samples  $x$  from class  $y_i$ ,  $i = 1 \dots K$ .

The importance of the feature can be measured using the mutual information  $MI(Y, X)$  between the target and the feature distributions is larger. Decision trees use similar metric known as Information Gain.

$$IG(Y, X) = H(Y) - H(Y|X) \quad (3.6)$$

It is the difference between information in the class distribution  $H(Y)$ , and the conditional information [48].

There is no definite answer as to which relevance index is the best. In many applications simple methods may be chosen, such as the correlation coefficient. The filter method is independent of the algorithm and works very well in high dimensional settings. It is fast and computationally efficient. Its disadvantage is

that it disregards the interaction between classifiers while most proposed techniques will consider each feature separately.

### 3.3 Wrapper

Identifying the best subset of features is very important for the model. While filter based methods can be used as discussed previously, however it fails to take into account the interaction between the classifiers. Therefore, if the evaluation method used employs the predictor architecture for which the features are chosen, the wrapper approach would be more suitable. Wrappers use the "black box" assumption to give a score to the feature subset depending on their predictive capability. The wrapper method effectively uses the inductive algorithm to perform evaluation. Cross-validation is a common wrapper based approach.

This method searches for features which are suited for the learning algorithm and aims to improve the model performance. Feature evaluation is done using the accuracy for classification. Moreover, the goodness of cluster is evaluated using clustering[49]. At the heart of the wrapper methods is the search strategy of all promising feature subsets. Subsets of features are thereof generated and evaluated accordingly.

An exhaustive search strategy can be used to find the optimal subset. Nevertheless, it is highly computationally inefficient. Hence, it is not feasible and rarely done. The Branch and Bound search strategy is another way of searching for subset of attributes. The strategy is based on the fact that once a subset  $S$  consisting of more than  $d$  variables has been evaluated that no subset of it can be better. This is due to its monotonic nature. Hence the subsets of  $S$  do not need to be evaluated.

Sequential selection is suboptimal search strategy introduced as a result of the infeasible methods discussed above. Sequential pruning was proposed by Marill and Green [50]. This will be referred to as sequential backward selection. It initializes the variable set with all features. Variables remaining in the set are assumed to be pruned. The results of the 'after excluding each variable' are obtained and compared using the evaluation function  $J(\cdot)$ . It ends by pruning the variable whose removal leads to having the best result. The pruning continues taking place until a prespecified number of variables is left, or until the results become very bad.

Sequential growing is another sub optimal search strategy and is similar to the sequential selection but adds variables at every step. We will refer to this as sequential forward selection. Initially each candidate feature that is not yet part of the current set is included into the current set, and the resulting set is evaluated. The variable whose inclusion has the best evaluation is inserted in the current set. It continues doing the same until a pre-specified number of variables is arrived at, or until no further improvement is shown. Typically, this method is faster than the sequential backward selection.

Several extensions to the main sequential selection processes are proposed. These include generalized sequential selection. The adding or removal of a set of  $g$  variables at a time is evaluated. When there are  $nk$  candidate features left to be included (excluded). The algorithm does not require as many steps as SFS and SBS.

Another extension is the Backtracking during search method. The objective of introducing this method is to counter the nesting effect. The nesting effect occurs since the feature that was added or removed can not be added or removed again later. Hence, the nesting effect occurs as bad decision done initially can

not be adjusted. Every step in this approach is split into two steps. In the first sub-step, SFS is run to include  $k$  variables while the next sub-step implements SBS to remove or exclude  $r$  features. Thus, the steps are reduced to smaller steps and this lowers the overall complexity.

The extensions discussed above are considered greedy as they seek the best subsets among the candidates. There might be certain branches missed where these branches could be useful to the overall result. The Beam search method maintains a list of the highly important branches. If more subsets are allowed to keep, then the beam search methods continues searching

The importance of the feature is measured by using machine learning algorithm and applied to each subset feature. The wrapper takes into consideration the interaction between feature subset search and model selection. The disadvantage is that it has a higher risk of overfitting and is more computationally demanding.

### **3.4 Embedded**

The embedded method doesn't separate learning and attribute selection, as is the case with filter and wrapper method. Feature selection can be interpreted as finding the feature subset that leads to the minimum risk. A binary vector is used to model the subset. Thus a feature that is in a subset has its indicator set to 1, and a feature that is absent has its indicator set to 0. The aim is to find a vector of indicator variables that minimized the expected risk or error.

Embedded methods reflect a form of regularization. They employ a constraint on the space dimensions. The predictors are parameterized depending on how much the output is affected by them.

Given a set of parameter weights  $\theta$ , the model for Linear classifiers is defined by:  $g_\theta(x) = \text{sign}(\theta^T x)$  or in Linear regression defined by  $g_\theta(x) = \theta^T x$ .

Hence, the following equation represents the  $L_0$  norm optimization.

$$\min_w \text{Loss}(g_\theta) + \lambda \|\theta\|_0 \tag{3.7}$$

In this equation,  $\|\theta\|_0$  returns the count of non-zero weights[51]. Hence, minimizing  $\theta$  induces a sparse feature set with few non-zero  $\theta_i$ 's and a model. The embedded feature selection performs the selection of attributes within the inducer. Due to the nature of the regularizer, the solution obtained is sparse.

The equation 3.7 has an exponential complexity and is infeasible. Therefore, the  $L_0$  norm is in fact approximated by the  $L_1$  norm. The  $L_1$  norm is important as it provides a sparse solution and gives way to a convex optimization problem so a global optimum is certainly obtained. Furthermore, sparse linear models may be proposed in the Bayesian context. This is referred to as Relevance vector machine proposed by Tipping which can be used for classification and regression as well. The Lasso regression method is another example that can be used as well.

## Chapter 4

# Regularized Logistic Regression for Yield Analysis

The objective of this chapter is to propose a fast logistic regression based importance sampling methodology with ordered feature selection to avoid overfitting and enable regularization. We rely on the importance region search simulations to build a regularized logistic regression model that is capable of accurately predicting pass fail criteria for purposes of yield analysis stage. We also propose a cross-validation-based regularization framework for ordered feature selection. We later prove the efficiency of the proposed methodology by analyzing state-of-the-art FinFET SRAM designs. The proposed methodology is comprehensive and computationally efficient resulting in high-fidelity models. This in turn translates into accurate yield prediction for the rare fail events. All this comes at significant savings in runtime. First, we start by giving an overview about some important concepts related to logistic regression.

## 4.1 Motivation

The past decade witnessed extensive utilization of logistic regression. Its applications were not strictly limited to engineering problems but also business analytics, the field of medicine, and social sciences [52, 53]. In the context of semiconductor manufacturing, logistic regression was used to model the fab measurements in terms of the “binary response for when a chip does or does not have bit failures” [54] as function of the recorded bit fail categories. Logistic Regression was mainly developed as a machine learning tool used for the prediction of faults. It is used to define a model that best represents the relationship between a categorical response and a given set of variables [55]. The estimated response or outcome of the logistic regression is usually binary. Regression be considered as a supervised learning approach having a discrete or continuous output. In fact, many statistical modeling techniques are regression based such as ordinary least squares (OLS) regression and ridge. Logistic regression utilizes the logistic function to map the outcome of a linear regression into a class label. The model built estimates the class probability while aiming at maximizing the probability of correct classification. In the context of memory yield analysis, we rely on logistic regression to capture the binary nature of SRAM fails and speed the circuit simulations of the importance sampling phase as will be explained later in this chapter.

## 4.2 Logistic Regression Overview

### 4.2.1 Linear Regression

The past decade witnessed intense interest in the utilization of regression. When the data is continuous, linear regression is typically selected to model the outcome.

In regular linear regression, a function  $f(x)$  is modelled as a linear combination of the independent variables vector  $X$  [17] according to equation 4.1.

$$f(X) = \sum_{i=1}^D \theta_i * x_i + \theta_0 \quad (4.1)$$

where  $\{\theta_i, i=1,2,3,\dots,D\}$  are the model coefficients,  $\{\mathbf{X} = x_i, i=1,2,3,\dots,D\}$  represents the  $D$ -dimensional variability vector, and  $\theta_0$  represents the constant term. Given a training set of  $N$  sample points with  $N > D$ , we define:

$$X = \begin{bmatrix} x_1^{(1)} & x_1^{(2)} & \dots & x_1^{(N)} \\ x_2^{(1)} & x_2^{(2)} & \dots & x_2^{(N)} \\ \vdots & \vdots & & \vdots \\ x_D^{(1)} & x_D^{(2)} & \dots & x_D^{(N)} \end{bmatrix} \quad (4.2)$$

and  $F = [f^{(1)} f^{(2)} \dots f^{(N)}]$  where  $f^{(j)}$  and  $x^{(j)}$  represent the value of  $f(x)$  and  $x$  at the  $j^{th}$  sample point respectively.  $N$  is usually greater than  $D$ , and there will be an overdetermined system of equations to find the best  $\theta$  using the training data. The hypothesis function  $h(\theta)$  is the resulting linear regression approximate model to  $f(x)$  as presented in 4.3, where  $x_0$  represents the intercept term. Ordinary Least Squares is used to find  $\theta$  and minimize the error between the obtained hypothesis function  $h(x)$  and  $f(x)$  for the training data [14].

$$h_{\theta}(\mathbf{X}) = \sum_{i=1}^D \theta_i * x_i \quad (4.3)$$

Hence, the cost function becomes:

$$J(x, \theta) = \frac{1}{2N} \sum_{j=1}^N (h_{\theta}(x^{(j)}) - f^{(j)})^2 \quad (4.4)$$

The least Mean Squares (LMS) update rule can be used to find a solution for  $\theta$  that minimizes the cost function. Table 4.1, presents the batch gradient decent approach.  $\alpha$  represents the learning rate, and  $\theta^0$  represents the initial guess for  $\theta$ .

Table 4.1: Pseudo Code for batch gradient descent [2].

Line #	Code
1.	Initialize $\theta = \theta^0$
2.	$\theta_i^{k+1} = \theta_i^k + \alpha \sum_{j=1}^N (h_{\theta}(x^{(j)}) - f^{(j)}) x_i^{(j)}$

The LMS method is foundational for approaches that aim at solving for the model parameters in other types of regression including the logistic regression. It should be also noted that for linear regression, a closed form solution of  $\theta$  that minimizes  $J_{\theta}$  exists and can be derived according to (5).

$$\theta = (X^{(T)}X)^{-1}X^{(T)}Y \quad (4.5)$$

To avoid overfitting, and enhance the model's accuracy the regularized linear regression attempts to solve for a penalized form of the cost function (4.4) [15], where  $l_p = 0, 1, \text{ or } 2$ . Thus, there are three main forms of regularization that are usually used. These include the  $L_0$ -norm,  $L_1$ -norm, and  $L_2$ -norm regularization.

The  $L_0$ -norm ensures sparseness by realizing the optimal subset of independent variables. On the other hand, it remains NP-hard. The  $L_2$ -norm which is simpler, minimizes the magnitude of the parameters. However, it does not ensure sparse solution in any way.

$$J(x, \theta) = \frac{1}{2N} \sum_{j=1}^N (h_{\theta}(x^{(j)}) - f^{(j)})^2 + \lambda \|\theta\|_{l_p} \quad (4.6)$$

## 4.2.2 Likelihood and Maximum Likelihood Estimation

In this section, we provide the definition of likelihood, as it will be later considered when studying the metrics for model selection. For a linear regression problem, we assume that the sample points are independent and identically distributed, and that the corresponding error vector follows a multivariate Gaussian distribution conditional on  $x$ , then the following relations in 4.7 holds.

$$f_Y(y_i|X) = (\mathbf{2}\pi\sigma_0^2)^{-\frac{1}{2}} \exp\left(-\frac{\mathbf{1}(y_i - x_i\beta_0)^2}{\sigma_0^2}\right) \quad (4.7)$$

Where  $y_i$  represents one sample point, and  $y_i - x_i\beta_0$  represents the error term whose variance is known to be  $\sigma_0^2$ . This relation can be generalized to multi-dimensional linear regression. The likelihood represents how plausible is a set of model parameter values, given the set of observed data. For the simple 1-dimensional linear regression problem, the likelihood can be expressed according to Equation

$$L(\theta, \sigma_0^2; y, X) = (\mathbf{2}\pi\sigma_0^2)^{-\frac{N}{2}} \exp\left(-\frac{\mathbf{1}\sum_{i=1}^N (y_i - x_i\beta_0)^2}{\sigma_0^2}\right) \quad (4.8)$$

The parameters  $\theta$  that maximize the Likelihood function correspond to those that minimize the sum of squared error. Hence, they coincide with the simple OLS solution under the assumptions above.

## 4.2.3 Logistic Regression

Logistic regression, being a popular machine learning algorithm borrowed from statistics, gained widespread attention and increased utilization in the past decade. It has been employed not only in engineering problems but also business analytics,

the field of medicine, and social sciences [56, 57]. In the context of semiconductor manufacturing, logistic regression was used to model the fab measurements in terms of the binary outcome representing a chip exhibiting or not exhibiting bit failures” [58] as function of the recorded bit fail categories. Logistic Regression was mainly developed as a statistical analytical tool used for fault prediction. Logistic regression is mainly used to build a model that best describes the relationship between a categorical response and a given set of input variables [59]. The model is indeed is not a straightforward classifier. The output of the model represents a probability which is in turn used to determine the outcome. The final outcome approximated by the logistic regression based model is usually binary. To enable this, the logistic regression hypothesis function,  $h(x)$ , therefore, returns a probability value that determines whether the output is 0 or 1 as indicated in equations 4.9 and 4.10 [59].

$$h_{\theta}(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}} \quad (4.9)$$

$$y = \begin{cases} 1 & h_{\theta(x)} > 0.5 \\ 0 & h_{\theta(x)} \leq 0.5 \end{cases} \quad (4.10)$$

$g(z)$  is the sigmoid function which returns a probability value that is typically between 0 and 1,  $x$  is a feature vector of size  $(n+1)$ ,  $\theta$  is a weight vector that has the length  $(n+1)$ . The value of the weight vector is chosen such that it maximizes the log-likelihood function as follows.

1. The probability of a given sample point given  $\theta$  is represented by equations (4.11, 4.12) below.

$$\begin{cases} p(y = 1|x; \theta) = h_{\theta(x)} \\ p(y = 0|x; \theta) = 1 - h_{\theta(x)} \end{cases} \quad (4.11)$$

$$p(y_i|\theta) = h_\theta(x)^{y_i} (1 - h_\theta(x))^{(1-y_i)} \quad (4.12)$$

2. The likelihood function for all the sample points is the product of the singular sample point probabilities across all the sample points.

$$L(\theta; y, X) = \prod_{i=1}^N p(y_i|\theta) \quad (4.13)$$

3. As such, we solve for  $\theta$  that maximizes the log-likelihood function derived from (4.13) as indicated in (4.20).

$$l(\theta) = \sum_{i=1}^N \log(h_\theta(x^{(i)}) \cdot y^{(i)}) + \log(1 - h_\theta(x^{(i)})) \cdot (1 - y^{(i)}) \quad (4.14)$$

This would be similar to minimizing a cost function  $J(x; \theta) = -l(\theta)$ , which is also defined as follows in literature.

$$J(\theta) = \frac{1}{N} \sum_{i=1}^N (Cost(h_{\theta(x)}, y)) \quad (4.15)$$

$$Cost(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases} \quad (4.16)$$

No closed form solution exists for  $\theta$ . The solution can be obtained using the Newton's method or the stochastic gradient descent rule as shown in equation 4.17 [60]. This method behaves in a similar manner to the batch ascent rule in linear regression. The update is applied to each sample point, the function  $h_\theta$  is not linear, and it is based on  $\frac{\delta}{\delta(\theta_j)}l(\theta)$ . This somehow

resembles the LMS method.

$$\theta_j^{k+1} = \theta_j^k + \alpha \left( y^{(i)} - h_\theta(x)^{(i)} \right) x_j^{(i)} \quad (4.17)$$

#### 4.2.4 Logistic Regression Alternate Formulation and Model Building

Logistic Regression was mainly developed as a statistical analytical tool used for fault prediction. Logistic regression can be used to build a model that describes the relationship between a categorical response and a given set of input variables [58]. Thus, given  $N$  sample points for a  $D$ -dimensional variable vector  $\mathbf{x} = \{x_j; j \in [1, D]\}$ , and a categorical response  $y$  representing pass/fail criteria for a specific performance metric, the logistic regression hypothesis function,  $h_\theta(X)$ , returns a probability value that determines whether the output is 0 or 1 as indicated in equations (4.18) and (4.19) below [58].

$$h_\theta(\mathbf{x}) = g(\theta^T \mathbf{x}) = \frac{1}{1 + e^{-\theta^T \mathbf{x}}} \quad (4.18)$$

$$y = \begin{cases} 1 & h_\theta(\mathbf{x}) > 0.5 \\ 0 & h_\theta(\mathbf{x}) \leq 0.5 \end{cases} \quad (4.19)$$

$g(z)$  is the sigmoid function which returns a probability value that is typically between 0 and 1, and  $\theta$  is a  $D$ -dimensional weight vector whose value is chosen such to maximizes the log-likelihood function.

$$l(\theta) = \sum_{i=1}^N \log(h_\theta(\mathbf{x}^{(i)}) \cdot y^{(i)}) + \log(1 - h_\theta(\mathbf{x}^{(i)})) \cdot (1 - y^{(i)}) \quad (4.20)$$

The function  $h_\theta$  is not linear, and no closed form solution exists for  $\theta$ . several optimization methods can be utilized to solve for the maximizing a-posteriori parameter estimate. These methods typically rely on Newton's method or the stochastic gradient descent rule in an iterative manner.

The authors in [61] proposed a Newton based Iterative Reweighted Least Squares (IRLS) method to efficiently solve for the logistic regression model parameters. Their formulations relies on the categorical function  $y \in [-1, +1]$  and the following alternate representation of the likelihood function:

$$\min_{\theta} \sum_{i=1}^N -\log p(y^{(i)} | \mathbf{x}^{(i)}; \theta) \quad (4.21)$$

where  $p(y^{(i)} | \mathbf{x}^{(i)}, \theta) = g(y^{(i)} \theta^T \mathbf{x}^{(i)})$  Their approach relies on three key elements:

1. The gradient,  $\text{grad}$

$$\text{grad}(\theta) = \nabla_{\theta}(l_{\theta}) = \sum_{i=1}^N (1 - g(y^{(i)} \theta^T \mathbf{x}^{(i)})) y^{(i)} \mathbf{x}^{(i)} \quad (4.22)$$

2. The diagonal matrix  $\Lambda$ , with diagonal elements  $\Lambda_{ii}$  defined as function of  $\theta^{(k)}$ , where  $\theta^{(k)}$  is the solution of  $\theta$  at iteration  $k$  as will be explained next.

$$\Lambda_{ii} = g(\theta^{(k)T} \mathbf{x}^{(i)}) (1 - g(\theta^{(k)T} \mathbf{x}^{(i)})) \quad (4.23)$$

3. The Hessian matrix  $H$  which is given by the following equation, where  $X$  is an  $(N \times D)$  matrix as illustrated below.

$$H(\theta) = -X \Lambda X^T \quad (4.24)$$

$$X^T = [\mathbf{x}^{(1)} \mathbf{x}^{(2)} \dots \mathbf{x}^{(N)}] \quad (4.25)$$

At every iteration, the approach solves for a step direction by relying on Taylor series expansion as a quadratic approximation to the logistic regression cost function. Thus, assuming that  $\theta^{(k)}$  represents the current solution in the  $k^{th}$  iteration, the closed form solution for the step direction can be obtained as follows [61].

$$\gamma^{(k)} = \theta^{(k)} + H^{-1}\theta^{(k)}grad(\theta^{(k)}) \quad (4.26)$$

The solution for  $\theta^{(k+1)}$  can then be obtained via Newton's method via a line search according to [62], where  $t$  is a step size parameter that can be found via line search as will be explained later.

$$\theta^{(k+1)} = (1 - t)\theta^{(k)} + t(\gamma^{(k)}) \quad (4.27)$$

Relying on equations, (4.22), (4.24) and (4.23), it has been shown that the solution for the step direction,  $\gamma^k$ , is equivalent to that of solving a weighted least squares (LSQ) problem. By substituting  $H(\theta^{(k)})$  from (4.24) and  $grad(\theta^{(k)}) = X\Lambda(z - X^T\theta^{(k)})$ , where,  $z$  is defined in (4.28), Equation (4.26) becomes (4.29).

$$z^{(i)} = \mathbf{x}^{(i)T}\theta^{(k)} + \frac{[1 - g(\mathbf{y}^{(i)}\theta^{(k)T}\mathbf{x}^{(i)})]y^{(i)}}{\Lambda_{ii}} \quad (4.28)$$

$$\gamma^{(k)} = (X\Lambda X^T)^{-1}X\Lambda z \quad (4.29)$$

which in turn is the solution for the LSQ problem stated below.

$$\gamma^{(k)} = \arg \min_{\gamma} \|(\Lambda^{\frac{1}{2}}X^T) \gamma - \Lambda^{\frac{1}{2}}z\|_2^2 \quad (4.30)$$

### 4.3 Overfitting and Regularization Regression

William of Ockham, in the 14th century, first made "the law of briefness" well known. He wrote in Latin: "More things should not be used than are necessary" [63]. Overfitting is when the statistical model has a large number of parameters more than what is really needed to represent the data. A highly overfitted model, can provide a good training error since all data points are satisfied. However, when applied to test data, the error is typically higher for an overfitted model since it over-interprets the data. In the example of the figure presented below, the model simply passes through all training data points. Underfitting is when the parameters do not completely represent the data, as is the case with the average model presented below. Regularization is used to enhance the accuracy of the model and address the overfitting problems. Regularized regression works out a "penalized" cost function for the purpose of identifying the proper parameter vector. There are three main forms of regularization:  $L_0$ -norm,  $L_1$ -norm, and  $L_2$ -norm regularization. In the case of the ordinary linear regression, where

$$\hat{y} = \sum_{j=1}^n \theta_j * x_j \quad (4.31)$$

$L_p$ -norm regularization is of the form 4.32, where  $l_p=0, 1$ , or  $2$ .

$$\min_{\theta} \frac{1}{m} \sum_{i=1}^m (y_i - \hat{y}(x, \theta))^2 + \lambda \|\theta\|_{l_p} \quad (4.32)$$

$L_0$ -norm attempts to find the optimal subset of selected features required for the model; it ensures sparseness, however the formulation is NP-hard [64]. The  $L_1$ -norm intends to zero out weights, i.e., induce sparseness, and can be approximated. The  $L_2$ -norm optimization problem is simpler and minimizes the

amplitude of the weights; however, it does not enforce sparseness. For logistic regression,  $J(\theta)$  becomes for  $l_p=2$ :

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m (\text{Cost}(h_{\theta(x)}, y)) + \lambda \sum_{j=1}^n (\theta_j)^2 \quad (4.33)$$

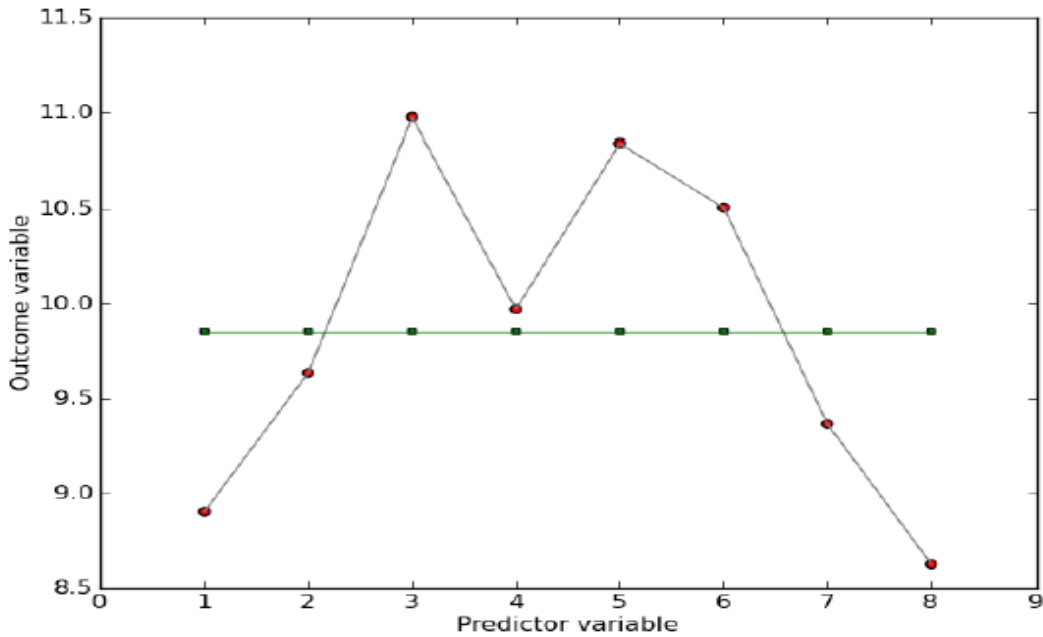


Figure 4.1: : A simplified illustration of (a) overfitting: model passes through all training points requiring a large number of parameters parameters, versus (b) underfitting: a single parameter (the mean) is used to build the model[3]

## 4.4 Proposed Regularization Framework for Fast Importance Sampling

The typical Importance Sampling flow relies on two phases. The first employs uniform sampling to search for the center of gravity and the second generates importance samples to estimate the yield. Both stages employ circuit simulations to evaluate the performance metric for the design understudy. Those two stages can

be summarized by the highlighted blocks of Figure 4.2. In the proposed approach we rely on regularized logistic regression to build a model for the design metric which can be used to predict the performance metric for the importance sample stage and hence avoid extensive use of circuit simulations. This is summarized in the dashed block of Figure 4.2. The figure thus presents an overview of the flow diagram of the proposed methodology which is based on two major steps. For the first step, we normalize the quadratic feature vector for phase 1 data, and build the corresponding regularized logistic regression model to obtain the corresponding model parameters  $\theta$  crit. For our regularization we rely on cross-validation and ordered selection of the feature vector as will be discussed next. The developed model is then used to estimate the response for phase 2 sample points eliminating the need for further circuit simulations. Note that the phase 2 data subspace is a subset of phase 1 data space as the uniform data varies all the parameters between  $\pm 6$  standard deviations. Accordingly, the model built using phase 1 should be applicable to phase 2 data.

## 4.5 Proposed Regularization Framework-Heuristics Based Approach

Traditionally feature selection is performed via forward, backward, or subset selection methodologies. Figure 4.3 presents the flow for a forward selection model building approach. Others have relied on injecting non-relevant random features to aid in the feature selection. For Linear regression  $L_1$  and  $L_0$  norm regularization are utilized. In order to eliminate the non critical feature weights, we relied on heuristics to solve the otherwise complex  $L_0$ -norm regularizations. In [62], the authors convert the  $L_1$ -norm regularized logistic regression to an

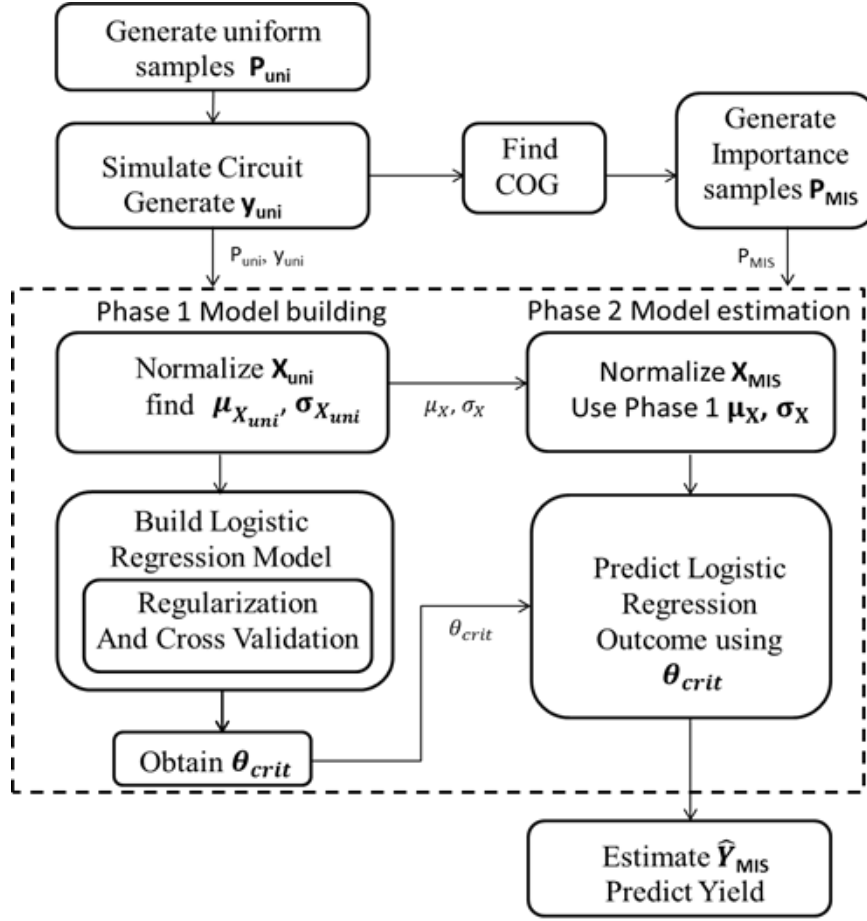


Figure 4.2: Methodology Flow Diagram [1]

$L_1$ -constrained iterative least square problem. We will elaborate more on these approaches in the following sections. The authors in [65] rely on local derivatives to identify features which lead to cost reduction that outweighs the penalty of the feature weight.

Herein, we introduce the following two-step heuristic approach based on ordered feature selection for the model building in phase 1 as illustrated in Figure 4.4 and stated below.

1. Initialization and Building: In this step, we build the initial model based on logistic regression. Order the normalized feature criticality based the

## Forward Selection

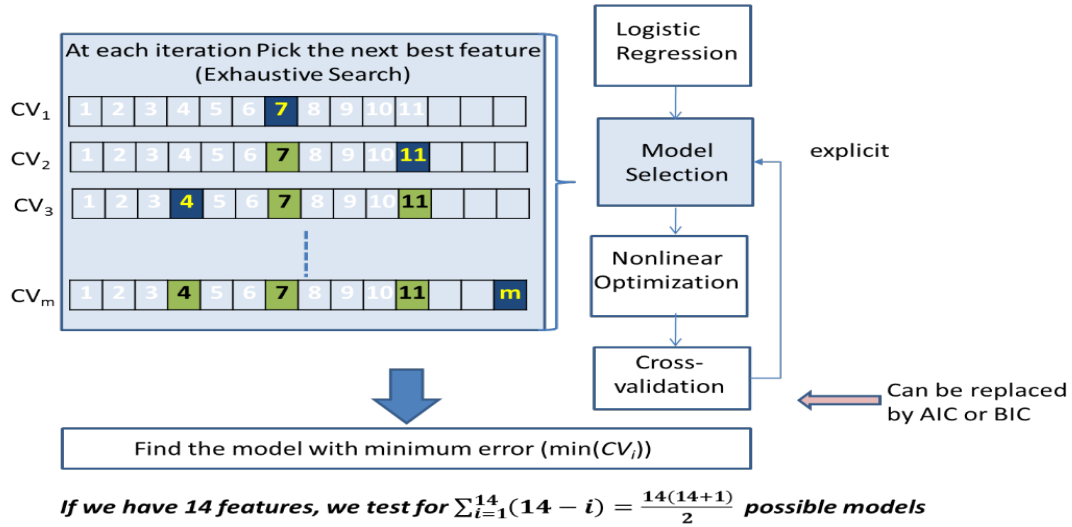


Figure 4.3: Basic feature selection approach.

## Heuristic Approach

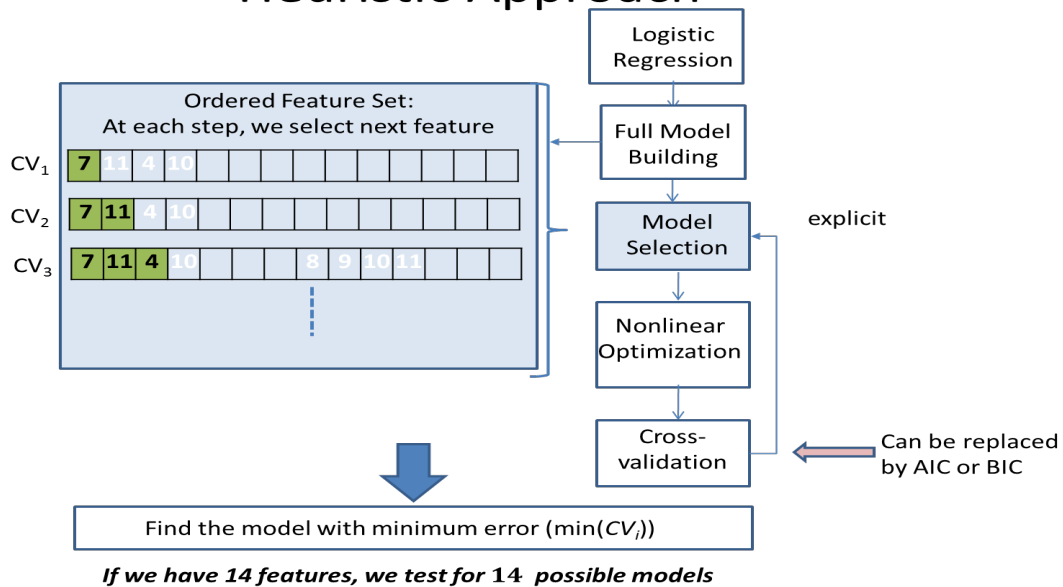


Figure 4.4: Proposed Heuristic Approach.

resultant  $\theta$  value.

2. Cross-validation based Search: In this part, for every iteration  $j$ , we use the top  $j$  critical variables based on the ordering step 1. We build a logistic regression model without regularization and track the test set error. We find the best set of critical features based on iteration corresponding to the minimum cross-validation error.

The pseudo-code for the proposed methodology is presented below. We normalize data for model building and feature ranking. We perform cross validation over  $j$  most critical variables to identify the best feature vector and corresponding  $\theta_{crit}$ . Finally, we use  $\theta_{crit}$  to predict the second phase  $\hat{y}_{MIS}$  and compute the yield using  $\hat{y}_{MIS}$ . We define three sub functions.

1. quadMap: represents the transformation as illustrated in 4.34, where  $p_{ij}$  represents a process variation parameter,  $i$  represents the sample number, and  $j$  represents the feature index; quadMap results in linear, quadratic and interaction terms.

$$X = \begin{bmatrix} p_{11} & p_{21} & \cdots & p_{11}^2 & p_{21}^2 & \cdots & p_{11}*p_{21} & \cdots \\ p_{12} & p_{22} & \cdots & p_{12}^2 & p_{22}^2 & \cdots & p_{12}*p_{22} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ p_{1m} & p_{2m} & \cdots & p_{1m}^2 & p_{2m}^2 & \cdots & p_{1m}*p_{2m} & \cdots \end{bmatrix} \quad (4.34)$$

2. buildLogistic: solves for according to 4.6.
3. predictLogistic: predicts  $y$  according to 4.10

---

**Algorithm 1** Yield Estimation Methodology [1]

---

**Parameters** Given process variation and response vectors  $P_{uni}$ ,  $F_{uni}$ , and  $P_{MIS}$

**Initialization**

- 1: Generate  $Y_{uni}$ ,  $X_{uni} \leftarrow quadMap(P_{uni})$  using 4.7, 4.8 and 4.9
  - 2: Generate mean  $\mu_{X_{uni}}$  and  $\sigma_{X_{uni}}$  for  $X_{uni}$
  - 3: Normalize uniform feature  $X_{n_{uni}} = \frac{X_{uni} - \mu_{X_{uni}}}{\sigma_{X_{uni}}}$
  - 4: Generate  $\theta_{init} = buildLogistic(X_{n_{uni}}, Y, \lambda)$
  - 5:  $\theta_{sort} \leftarrow sort(\theta_{init})$ , and  $I_{dx} | \{\theta_{sort,i} = \theta_{init} I_{dx,i}\}$
- 

*Phase 1 – Model Building: Phase1 data*

---

- 6: **for**  $j = 1 : size(\theta_{sort})$  **do**
    - Perform Cross Validation over j first critical vars
    - 7:  $X_{crit} \leftarrow X_{n_{uni}}(I_{dx}(1 : j))$
    - 8: **for**  $K = 1 : 4$  **do**
      - 9: Data sets:  $\{\mathbf{X}_{crit_{test}}, \mathbf{Y}_{test}\}_K, \{\mathbf{X}_{crit_{train}}, \mathbf{Y}_{train}\}_K$
      - 10:  $\theta_{K,j} \leftarrow buildLogistic(\{\mathbf{X}_{crit_{train}}, \mathbf{Y}_{train}\}_K, (\lambda = 0))$
      - 11:  $\mathbf{Error}_{K,j} \leftarrow predictLogistic(X_{crit_{test}}, \mathbf{Y}_{test}, \theta_{K,j})$
      - 12:  $\mathbf{Error}_j \leftarrow avg(\mathbf{Error}_{K,j})$
      - 13: **if**  $\mathbf{Error}_j < min$  **then**
        - 14:  $min = \mathbf{Error}_j$
        - 15:  $count = 0$
        - 16:  $numcriticalvars \leftarrow j$
        - 17:  $X_{crit} \leftarrow X_{n_{uni}}(I_{dx}(1 : j))$
        - 18:  $\theta_{crit} \leftarrow build_{logistic}(X_{crit}, Y, \lambda = 0)$
      - 19: **else**
        - 20:  $count = count + 1$
        - 21: **if**  $count > 4$  **then**
          - 22: **break**
        - 23: **end if**
      - 24: **end if**
    - 25: **end for**
  - 26: **end for**
- 

*Phase 2 – Importance Sampling and Yield Prediction (Phase 2 data)*

---

- 27: Generate  $X_{MIS} \leftarrow quadMap(P_{MIS})$  based on 4.7, 4.8
  - 28: Normalize to model  $X_{n_{MIS}} = \frac{X_{MIS} - \mu_{X_{uni}}}{\sigma_{X_{uni}}}$
  - 29:  $\widehat{y_{MIS}} = predictLogistic(X_{n_{MIS}}, \theta_{crit})$
  - 30:  $\widehat{Yield} \leftarrow Yield(P_{MIS}, \widehat{y_{MIS}})$
-

## 4.6 Cross Validation

Cross-validation is used as a means to estimate the model error. The average model error can then be used to select the best model as a means to avoid overfitting. Typically, when studying the model error; we split the data set into a training set and test set. In K-fold cross validation the given data set is split into K-sets. Regression is performed K-times. Each time regression is performed, K-1 sets are used for training to build the model, and one set is used to test the model error. The average error obtained from the K-errors is the overall modeling error [11]. There is a trade-off in the choice of K. The authors in [4] indicate that for real-word datasets the best method to use for model selection is ten-fold stratified cross-validation even if computation power allows using more folds. Typically, large K values help reduce the bias of overestimating the model error; this however, increases the runtime compared to small K values. On the other hand, small k values demonstrate reduced model stability due to the increased size of the perturbation in the data set [4]. For our purposes, we are using the estimated error to decide on the best model for purposes of model reduction. Hence, we are comparing among different models. For the sake of reduced runtime, we relied on 4-fold cross-validation as indicated in Fig.4.5. Other metrics exist in literature for proper model selection that do not require multi-fold simulations. These introduce significant savings in runtime. These include AIC/BIC and Mallows' Cp [66, 67, 68, 69]. Some include bias for logistic regression, and some were adjusted to reduce this bias. Exploring these metrics will be part of our future work as will be discussed next.

The Kull-back Liebler divergence can tell us the distance between a real distribution and a model distribution. The likelihood states the probability of a

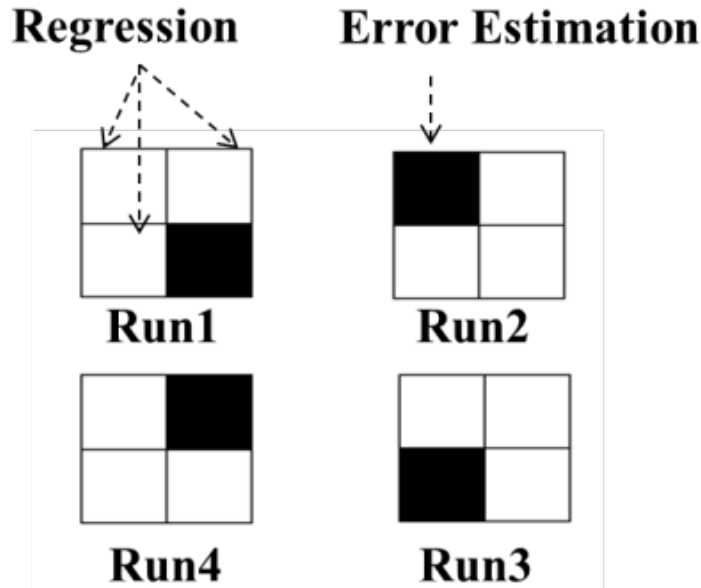


Figure 4.5: Example of 4-fold cross validation[4]

dataset given the model parameters. Since for most data sets we do not know the real life distribution/model, Akaike proposed in [66] a specialized metric, Akaike Information criterion (AIC), to extract information about how close a given model would be to the real unknown one compared to other competing models. In other means it is a measure of the quality of the model. Several criterion have been proposed since then in order to identify the best statistical model, such as BIC(Bayesian Information Criterion), SIC(Schwartz criterion), FIC, and DIC. The most popular though are AIC and BIC.

## 4.7 AIC BIC and Corrected AIC

AIC was first introduced in [66] as a means for determining the best models among a set of candidate models. The AIC was established by Akaike to provide an unbiased asymptotic estimator to the expected Kullback-Liebler information

that is used to compare two models even if they have different dimensions. His goal was to provide an approximation of the data lost when a statistical model is chosen to represent the model used to generate data. Akaike thus identifies an entropy measure to capture this info. The main principle derived based on the maximum log-likelihood expression in 4.35 [70].

$$E \left( \log \left( f(X|\hat{\theta}) \right) \right) = E \left( \int f(X|\theta) \log(f(X|\hat{\theta})) \right) \quad (4.35)$$

Where represents the set of estimate of the parameter  $\theta$  with probability density  $f(x|\theta)$ .

Where  $\hat{\theta}$  represents the set of estimate of the parameter  $\theta$  with probability density  $f(x|\theta)$ . AIC was very appealing to statistician since the concept relies on the maximum likelihood which is very amicable to statisticians. Enhancing the likelihood, however, may occur by overfitting, and AIC relies on a penalty to resolve this issue. In fact, AIC represents a form of penalized likelihood criteria as follows. AIC is minus twice a penalized log likelihood value as in equation : where L represents a maximum likelihood function over the parameters for a certain model, and k represents the dimensionality or cardinality of  $\theta$ .

$$AIC \{f(.,;\theta)\} = -2\ln \left( L(\hat{\theta}) \right) + 2 * \dim(\theta) = -2l_{max} + 2 * k \quad (4.36)$$

Where k is the number of parameters in the model. The best model is referred to as the model having the lowest AIC value. In general, AIC tends to overfit; this is based on simulations when the true model is known [71]. It is usually preferred for complex models yet less appealing for simple ones [68]. Bayesian information criterion, to be discussed next, puts a higher penalty, and accordingly it has been shown that it is better when the true model is available. Furthermore, a

corrected version of AIC,  $AIC_C$ , has been shown to be better than BIC in the absence of the true model [72]. For linear regression, it has been closely related to the leave-one-out cross-validation.

BIC: Bayesian Information Criterion or Schwartz information criterion is a model selection criterion. It is based partly on the likelihood and very similar to the previously discussed AIC. It can be computed as follows:

$$BIC = \ln(n)k - 2 \ln \left( L(\hat{\theta}) \right) \quad (4.37)$$

The BIC criterion has two main drawbacks:

1. The approximation is valid when the number of sample points is much larger than  $k$  which is the number of parameters in the model.
2. It has been shown by simulation that it can't handle complex collection of models as in variable selection. It behaves as an approximation to Bayes Factor which in turn is similar to the likelihood ratio test yet with the ability to account for a penalty term that represents the complexity of the model structure. [68]

Corrected AIC: It was later observed that the AIC has a tendency to overfit and several authors attempted to address this issue by proposing a more reliable or new version of the original Akaike criterion. Hurvish1991 [71] proposed bias correction. For a normal linear regression, or autoregressive,  $AIC_C$  criteria can be more generally defined as follows:

$$AIC_C = 2 \log L \left( \hat{\theta} \right) - \frac{2ndim(\theta)}{n - dim(\theta) - 1} = AIC - \frac{2dim(\theta)(dim(\theta) + 1)}{n - dim(\theta) - 1} \quad (4.38)$$

The authors in [73] propose another format of  $AIC_C$  that is more suitable

for logistic regression and is defined as shown below.

$$CAIC = -2l(\hat{\beta}; y) + 2r + \frac{1}{n}(\hat{a}_1 + \hat{a}_2 + \hat{a}_3) \quad (4.39)$$

Finally, it is worth noting, that all of these metrics and the Cross-validation methods help the selection to happen among a set of suggested models. With no guidance, one must exhaustively test all possible model options. With more computational power on hand, it is possible to do this exhaustive search for a large number of variables. Alternatively, the discussed metrics can be utilized to judge onto the best models via the proposed heuristic approach for parameter selection or guided L1-regularized solutions.

# Chapter 5

## Group LARS Iterative

## Reweighted Least Squares

## Methodology

Classifiers are the workhorse for many machine learning applications. Iterative reweighted least squares methods have been employed along with Least angle regression for efficient  $L_1$ -regularized logistic regression solutions. In this section, we propose an efficient  $L_1$  regularized logistic regression methodology with application to importance sampling based rare event estimation. At the core lies a Group LARS-based methodology that tracks Newton's step direction solution evolution from one round of the solution to another and employs weighted directions to efficiently solve for the underlying  $L_1$ -constrained iterative least squares problem. It thus exploits Group LARS inherent ability to handle groups of variables to benefit from the natural evolution of the solution as it searches for the best model and critical number of features. When applied to the yield modeling of a 14nm FinFET SRAM design, our results demonstrate significant speedup

for Group LARS iterations compared to pure LARS based approach very high accuracy compared to pure circuit-simulations based methodology.

Herein, we present a Group-LARS IRLS methodology for efficiently solving the  $L_1$ -regularized Logistic regression problem presented in (5.1).

$$\begin{aligned} \min_{\theta} \quad & \sum_{i=1}^M -\log p(y^{(i)}|x^{(i)}; \theta) \\ \text{s.t.} \quad & \theta_1 \leq C \end{aligned} \tag{5.1}$$

## 5.1 Least Angle Regression (LARS) Overview

Given a  $N$ -dimensional feature matrix  $X$ , and an  $N \times 1$  performance metric response vector  $F$ , the  $L_1$ -regularized least squares problem (LSQ) is:

$$\begin{aligned} \min_{\theta} \quad & \|X^T \theta - F\| \\ \text{s.t.} \quad & \|\theta\|_1 < C \end{aligned} \tag{5.2}$$

where  $\theta$  is the  $D$ -dimensional model parameter vector for the LSQ regression. Parameter  $C$  is typically unknown and finding its optimal value is expensive [74].  $k$ -fold cross-validation has instead been used along with LARS to solve the  $L_1$ -regularized problem and determine the best model with the minimum Cross-Validation Error (CVE) [75]. LARS first finds the most correlated variable with the response. It then finds the maximum step along the equiangular direction such that another variable has as much correlation with the residual as the current variable, and proceeds to identify the next critical variables accordingly. LARS hence recursively finds the solution upto a maximum number of features, and can be best described as follows:

1. Initialize residual vector  $\mathbf{r}_0 = F$ , the active set  $I_0 = \Phi$ , and the model

parameters  $\theta_j = 0$ , where  $j \in [1, D]$ .

2. Calculate the correlation vector  $\mathbf{c}_i = X_{I^c}^T * \mathbf{r}_{i-1}$
3.  $I_i = I_{i-1} \cup x_{best}$ ; add the feature with max correlation
4. Find the direction vector s.t.  $X_I^T X_I \mathbf{d}_i(I) = \text{sign}(\mathbf{c}_i(I))$
5. Calculate step  $s_i$  in function of  $(\mathbf{c}_i, X_I, \mathbf{d}_i)$  along  $d_i$
6. Update  $\theta_i = \theta_{i-1} + s_i * \mathbf{d}_i$
7. Update  $\mathbf{r}_i = \mathbf{r}_{i-1} - X_I * \theta_i$
8. Go to 2 until a max number of desired features is met

## 5.2 Revisiting IRLS-LARS

The authors in [62] handle the regularization by relying on LARS to solve equation an  $L_1$  constrained version of (4.30):

$$\begin{aligned} \min_{\gamma} \quad & \|\Omega\gamma - F\| \\ \text{s.t.} \quad & \|\gamma\|_1 < C \end{aligned} \tag{5.3}$$

where  $\Omega = \Lambda^{\frac{1}{2}} X^T$  and  $F = \Lambda^{\frac{1}{2}} z$ . Since  $\Omega$  is a scaled version of  $X^T$ , features selected by LARS on  $\Omega$  reflect respective features in  $X$ . Thus, for a desired number of features  $n$ , they

1. Set initial parameter vector  $\theta_n^{(0)} = 0$
2. Iterate until convergence on  $\theta_n^{(k)}$ 
  - (a) Update  $\Omega$ , and  $\nabla$  using (4.23) and (4.22)

- (b) At each step solve step direction  $\gamma^{(k)}$  in (5.3) using LARS upto  $n$  non-zero elements
- (c) Update  $\theta_n^{(k+1)} \sim (4.27)$  using backtrack method [76].

Thus, for fixed  $n$ , at each nonlinear iteration,  $k$ ,  $\gamma^{(k)}$  is solved for through LARS over  $n$  iterations of steps 2-8 of Section 5.1. This is repeated as we sweep  $n$  and solve for the nonlinear optimization in the effort to find the best model and critical number of features that minimize the CV error. As such, step 2b will be invoked many times resulting in updating  $\gamma^{(k)}$  via LARS ( $\sum_{i=1}^{mNF} i$ ) times each invoking  $i$  LARS iterations, where  $mNF$  is the the maximum number of desired features. This results in a quadratic complexity in  $mNF$  rendering the cross-validation expensive. We propose, herein, a Group-LARS based IRLS approach, IRLS-GroupLARS, for enhanced efficiency for the regularized Logistic regression problem.

### 5.3 Proposed Group-LARS based IRLS

Figure 5.1 presents the flow diagram for the proposed IRLS-GroupLARS. At the core, we employ an adjusted Group-LARS methodology [77, 78] to efficiently solve for the linearized objective function. We exploit GroupLars’s inherent ability to handle groups of features simultaneously in order to reduce the number of required direction vector solution iterations while solving for the nonlinear problem. First, we provide an overview of Group LARS.

#### Group LARS Overview

Given a linear regression problem,  $Y = \sum_{j=1}^J X_j \beta_j + \epsilon$ , where  $X_j$  represents an  $N \times p_j$  matrix corresponding to the  $j^{th}$  factor (group of variables) and  $\beta_j$

represents a parameter vector of size  $p_j$  and  $j \in [1, J]$ . This formulations presents itself as a representation of the simple linear regression problem  $Y = X\beta + \epsilon$  where  $X = (X_1, X_2, \dots, X_J)$ , and  $\beta = (\beta_1, \beta_2, \dots, \beta_J)$ . Group LARS enables selecting important factors or groups of variables. Similar to LARS, all parameter vectors  $\beta$  are initially set equal to zero vector. When the factors have equal sizes, Group LARS initially finds  $X_j$  having the smallest angle/largest projection with  $Y$ ,  $\|X_j Y\|_2^2$ . It then proceeds in the direction of the projection of the response vector on the space defined by the factor  $X_j$  until, and determines a step along this direction such that some other factor has "as small angle" with the residual. When the factors have different sizes,  $p_j$ , Group LARS algorithm relies on the weighted projection of the residual on the factor  $\|X_j r\|^2/p_j$  for the factor selection [77, 78]. The IRLS-GroupLARS approach proceeds as follows.

1.  $critGroup = []$
2. for  $n = 1 : maxNumDesiredFeatures$ 
  - (a) Set  $\theta_{.,n}^{(0)} = 0$
  - (b) Iterate until converge on  $\theta_{.,n}$  solution
    - i. Update  $\Omega$ ,  $F$  and  $\nabla$  using (12)-(14)
    - ii. At each step solve step direction  $\gamma^{(k)}$  in (5.3) using  $GroupLARS(\Omega, F, critGroup)$  and find next best variable index  $j^*$
    - iii. Update  $\theta_{.,n}^{(k+1)}$  relying on back track line search method that determines a step update for the Newton method based on the Logistic regression cost function using  $\theta_{.,n}^{(k)}$ ,  $\gamma^{(k)}$ ,  $\nabla$ ,  $X$  and  $Y$  as indicated in Algorithm 3 at a given iteration.
  - (c) update  $critGroup = critGroup \cup j^*$

As such, we pass the critical group indices,  $critGroup$ , as a single factor, from one round of solving the nonlinear problem at a specified number of features  $n$  to another. At the end of each round,  $critGroup$  is appended with the next best variable. For example, at the end of round  $n - 1$ ,  $critGroup$  has the best  $n - 1$  variables. Thus, at round  $n$ , the proposed algorithm invokes only two iterations of the adjusted Group LARS in Algorithm 2 when solving for the step direction  $\gamma^{(k)}$  for a given  $\theta^{(k)}$ :

- In the first iteration, the algorithm handles  $critGroup$  as one factor, whose size is  $n - 1$  based on the previous round and determines its corresponding step  $s$  by determining the next factor,  $j^*$ , from the pool of factors of size one in ' $\Omega$ - $critGroup$ ' representing the remaining variables similar to lines 9-10 of Algorithm 2. Note that at this stage,  $\Omega$  and  $F$  represent the respective input features and response vectors. The algorithm, then proceeds to determine  $\beta$  for  $critGroup$  and updates the residual accordingly.
- In the second iteration, the algorithm finds the step,  $s$  for  $j^*$  from the previous iteration through identifying the next one dimensional factor in ' $\Omega$ - $critGroup$ - $j^*$ ' that has equal weighted correlation as  $j^*$ . It then updates  $\beta$  and returns it as the solution to the main algorithm step direction  $\gamma$  along with an updated factor group,  $I_l$ , corresponding to the union of  $critGroup$  and  $j^*$ .

The step direction  $\gamma$  is then used to solve for the linearized objective function, and the iterations per round proceed until we converge on the solution for  $\theta_{.,n}$ . Then, at the end of this round,  $critGroup$  is updated with the latest factor group returned from GroupLars to include the old  $critGroup$  and  $j^*$ . For the next round again, at each iteration when solving for  $\theta_{.,n+1}^{(k)}$ , we invoke Group LARS using the

new critGroup to solve for  $\gamma^{(k)}$  as discussed above.

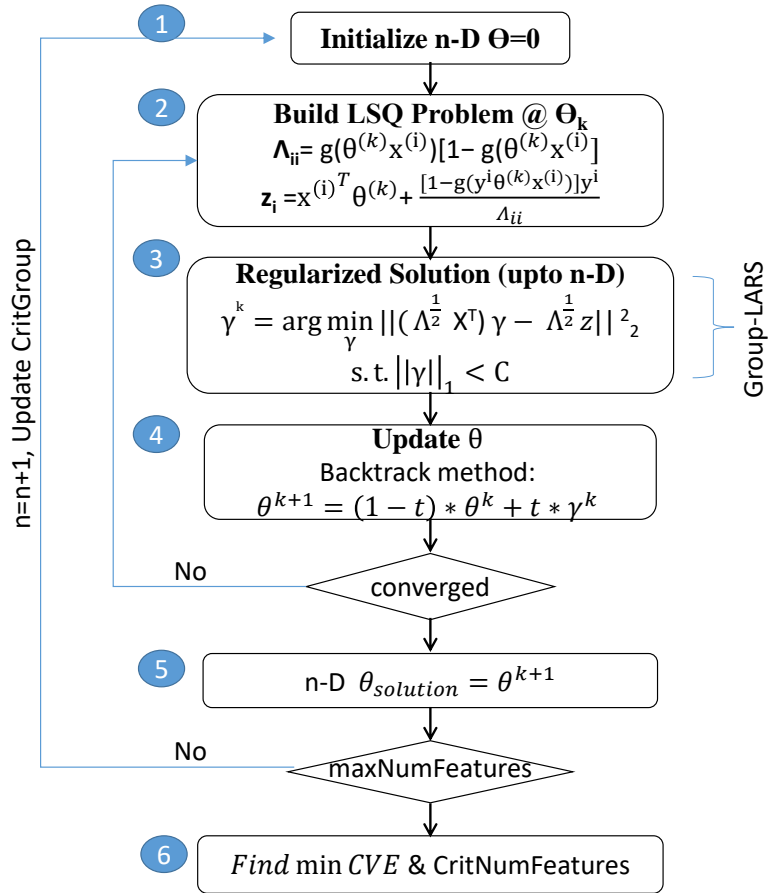


Figure 5.1: Methodology flow diagram [5].

---

**Algorithm 2** IRLS-GroupLARS [5]

---

**Input**  $X$ : Input Features,  $Y$ : Labels

**Output** Best model with lowest Cross-Validation Error

```
1: function MODGROUPLARS( $X, Y, critGroup$ )
2:   if  $critGroup = []$  then
3:      $critGroup = \max_j \|X_j^T Y\|_2^2$ 
4:   end if
5:   Initialize  $r_0 = Y, l = 1, I_l = critGroup, \beta_{l-1} = 0$ 
6:   Compute the least squares direction vector  $d$  with
7:    $d_{I_l^c} = 0$ , and
8:    $d_{I_l} = (X_{I_l}^T X_{I_l})^{-1} X_{I_l}^T * r_{l-1}$ 
9:   Find minimum step  $s^* = \min(s_j)$  where  $s \in [0, 1]$ 
   s.t. normalized squared correlation due to any of the active
   groups  $j' \in X_{I_l}$  equals the normalized squared
   correlation due to one of the groups  $j \in X_{I_l^c}$ 
10:   $\frac{\|X_j^T (r_{l-1} - s_j X_j \gamma)\|_2^2}{p_j} = \frac{\|X_{j'}^T (r_{l-1} - s_j X_j \gamma)\|_2^2}{p_{j'}}$ 
11:  Update  $I_{l+1} = I_l \cup \{j^*\}$ 
12:  Update  $\beta_l = \beta_{l-1} + s * d$ 
13:  Update  $r_l = r_{l-1} - X * \beta_l$ 
14:  if  $l == 1$  then
15:     $l = l + 1$ 
16:    Go to step 6 to find the next candidate
17:  else
18:    return  $I_l, \beta_l$ 
19:  end if
20: end function

21: // Begin Cross-Validation
22: Repeat Lines 25-37 for each CV iterations
23: Return number of features corresp. to min CVE error
24:  $critGroup = []$ 
25: for  $n = 2 : maxNumDesiredFeatures$  do
26:   Set  $\theta_{:,n}^{(1)} = 0$ 
27:   for  $k = 1 : maxNitters$  do
28:     Find  $\Lambda, \nabla$  and  $z \sim (4.23), (4.28), (??)$  using  $\theta_{:,n}^{(k)}$ 
29:     Set  $\Omega = \Lambda^{1/2} X^T, F = \Lambda^{1/2} z$ 
30:      $[I_n, \gamma^{(k)}] = MODGroupLARS(\Omega, F, critGroup)$ 
31:      $\theta_{:,n}^{(k+1)} = BackTrackLineSearch(\theta_{:,n}^{(k)}, \gamma^{(k)}, \nabla, X, Y)$ 
32:     Check for convergence of  $\theta_{:,n}^{(k+1)}$  to break
33:   end for
34:    $critGroup = I_n, \theta_{:,n} = \theta_{:,n}^{(k+1)}$ 
35:   return  $CVE_{(n,b)}$  // b is the CV iteration
36: end for
```

---

**Algorithm 3** Backtrack Line Search Method [5]

---

```
1: function BACKTRACKLINESEARCH( $\theta$ ,  $\gamma$ ,  $\nabla$ , X, Y)
2:   [N D]=size(X)
3:    $f_k = \sum \log(1 + e^{-Y\theta^{(k)T}X})$ 
4:   Define Constants  $c= 0.5$ ,  $\beta =0.8$  //  $\beta \in [0, 1]$ 
5:   Initialize  $t = 1$ 
6:    $\theta^{(k+1)}=\theta^{(k)}(1-t)+ t\gamma^T$ 
7:    $f_{k+1} = \sum(\log(1 + e^{-Y\theta^{(k+1)T}X})$ 
8:   while ( $(f_k + ct\nabla^T\gamma - \theta^{(k)}) < f_{k+1}$ ) do
9:      $t = \beta * t$ 
10:     $\theta^{(k+1)}=\theta^{(k)}(1-t)+ t\gamma^T$ ;
11:     $f_{k+1} = \sum(\log(1 + e^{-Y\theta^{(k+1)T}X})$ 
12:  end while
13: end function
```

---

# Chapter 6

## Data Imbalance Handling Approaches for Accurate Statistical Modeling

Data imbalance can impact the fidelity of a classifier. We rely on advances in data imbalance handling techniques for machine learning applications to propose an enhanced fast statistical analysis methodology. Particularly, we employ data handling techniques in the context of a logistic regression based importance sampling methodology for accurate statistical modeling of rare fail events in memory designs. We demonstrate that for purposes of achieving conservative yield estimates, the synthetic minority oversampling technique outperforms other data handling methods and portrays the best model recall and precision rates. We report significant reduction in the number of False Negatives compared to imbalanced data set based approaches. We also report on average a low relative error rate in the yield estimate for the balanced data set-based modeling approaches compared to the imbalanced data set-based approaches. These results

were verified on state-of-the-art industrial FinFET SRAM designs.

The objective of this chapter is to give a general overview about data imbalance and dive into some of the important techniques typically used to achieve this. A data balancing scheme is also presented and later on evaluated in the results section.

## 6.1 Motivation

Existing machine learning methodologies have achieved major successes in many fields. However, imbalanced datasets posed a major challenge for machine learning techniques. The imbalance problem was mainly encountered in areas such as fault diagnostics, fraud detection, and intrusion detection. An imbalanced dataset is characterized by having the number of points in one class not almost equal to that of another class [79]. The prediction accuracy of the model is affected by the imbalance. When using imbalanced data to learn a model, three main problems arise accordingly. These are improper evaluation metric for the model, small number of data points belonging to the minority class, and resultant weak classifiers [80, 81, 82]. There are two main types of approaches that have been proposed to tackle this issue and these are: algorithmic approaches and data approaches. Algorithmic approaches rely on performing adjustments at the heart of the algorithm. For example, improved boosting was an algorithmic approach proposed by [83]. Data approaches, on the other hand, utilize resampling techniques to pave way for balancing the classes before learning the model. These resampling techniques can be classified to undersampling and oversampling techniques. Undersampling methods aim at achieving lower sample points for the majority class whereas the oversampling method aims at having more sample

points in the minority class. Herein, we focus on 50/50 balanced data sets. Weiss and Provost [84] also discuss that the ideal class distribution depends a lot on the learning algorithm used. We will employ different ratios in our experiments in later chapters when evaluating SVM approaches. Herein, we start out discussion by discussing data based balancing techniques and its application to logistic regression-based SRAM yield modeling.

## 6.2 Data Balancing Techniques: Explicit

### 6.2.1 OverSampling Methods

Oversampling methods create new samples by oversampling the minority class. We discuss three main methods for oversampling.

Random OverSampling (ROS) is an oversampling approach that generates data by generating random copies of the original minority sample points. It implements oversampling of the minority class until there data points from the minority class as much as there are from the majority class [85].

Synthetic Minority Over-sampling Technique (SMOTE) is another popular oversampling method that was introduced in [86]. The method is particularly useful for applications targeting mining for rarity such as fraud detection[86]. For each sample point in the minority class,  $x_j$ , a new synthetic minority point is generated using  $x_j$  and one of its  $k$  minority nearest neighbors,  $x_n$ , based on a predetermined oversampling rate as shown in Figure 5.

Borderline-1 SMOTE is also another important oversampling methods. It aims at emphasizing the minority class sample points near the boundary region [87]. Hence, the ordinary SMOTE method is implemented only to borderline minority sample points. These are the class sample points whose nearest neighbors

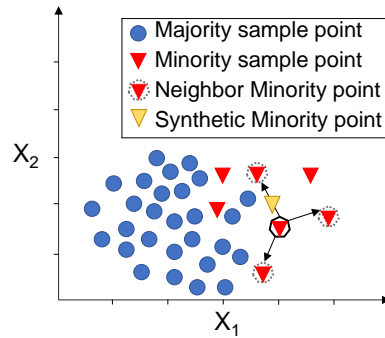


Figure 6.1: Synthetic sample point generation (golden triangle) or a specific minority sample point (bold line) using SMOTE [6].

include some majority points.

### 6.2.2 UnderSampling Methods

These methods aim at lowering majority class sample points. Random Under-Sampling (RUS) method randomly removes points from the majority class until the number of sample points is the same as that of the minority class [85].

The NearMiss-2 method is another undersampling technique that eliminates those points from the majority class that are nearest to the minority class [88]. This is determined based on an average distance metric that is computed with respect to the three farthest minority points. The method was shown to perform better than other versions.

## 6.3 Proposed Data Balancing Logistic Regression Based Method

Given the design performance metric  $f(x)$  and  $f_0$  is the corresponding critical value that sets the pass/fail threshold for the design, and  $I(x)$  is the indicator

---

**Algorithm 4** Smote Pseudo Code (adapted from [89])

---

```
1: procedure SMOTE( $X_{min}$ , N, k) {Returns  $\frac{N}{100} * T$   $x_{syn}$  sample points}  $\{X_{min}$   
   is the set of the T minority sample points to be smoted}  
2:   if  $N < 100$  then  
3:     Randomize the T sample points  
4:      $T = (N/100) * T$   
5:      $N = 100$   
6:   end if  
7:    $N = (int)(N/100)$   
8:    $X_{syn} = \{\}$   
9:   for  $i = 1$  to  $T$  do  
10:    find k nearest minority neighbor points  $\{X_{nn}\}_i$  for the  $i^{th}$  sample point  
11:     $X_{syn} \leftarrow X_{syn} \cup \text{GenerateSamplePoints}(N, i, \{X_{nn}\}_i)$   
12:  end for  
13: end procedure
```

---

function defined below.

$$I(x) = \begin{cases} 0, & \text{pass } (f(x) < f_0; x \in \mathbb{R}^n) \\ 1, & \text{fail } (f(x) > f_0; x \in \mathbb{R}^n) \end{cases} \quad (6.1)$$

In the event of rare fail event problems, Monte Carlo needs an infeasible large number of sample points to capture the fail probability accurately. Importance sampling as discussed in chapter 2 relieves this issue by distorting the natural distribution to bias the sampling more towards the fail region [90].

Figure 6.2 presents the proposed methodology flow diagram. For Importance Sampling, uniform sampling is employed in the initial phase to determine the importance sampling distribution and fit the logistic regression model. The proposed approach uses a regularized logistic regression model in the importance sampling second stage to accelerate the simulations as in [1]. It exploits the uniform sampling stage to build this model. In the field of memory design, rare fail events are of particular interest. It appears that the number of failing points in

the uniform sample stage is relatively small as compared to the total number of samples as will be illustrated in Figure 6.3 and 6.4. Importance sampling shift center computation is unaffected by this, since it is computed as the center of gravity (mean) of the failing points. Typically a few fail points are enough to calculate the center of gravity. However, this class imbalance originally presented in the dataset can impact the model performance and disrupt the logistic regression-based classification. As a result, the final yield estimate will also be affected. In fact, Figure 6.3 shows a theoretical example that is further complemented by real circuit simulations in Figure 6.4 and together both figures show that there is clear imbalance between the number of majority passing and minority failing sample points created in stage 1 for rare fail events. Furthermore, Figure 6.4 shows that the ratio of minority samples to the total number of sample points, obtained in the first phase, can drop to few percent for designs with yield values in range of 4-6 sigma. As such, the proposed methodology employs a data balancing scheme that can enhance the emphasis of the model on the minority class without affecting the majority. Our aim is to improve the recall rate of the model, i.e., the ability of capturing all fails correctly while maintaining good precision.

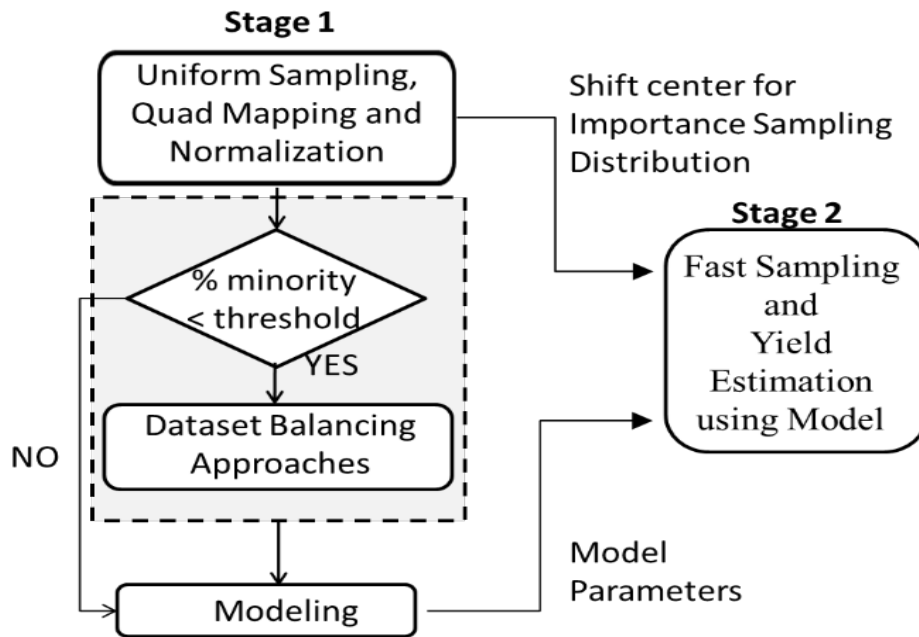


Figure 6.2: Methodology Flow Diagram [6].

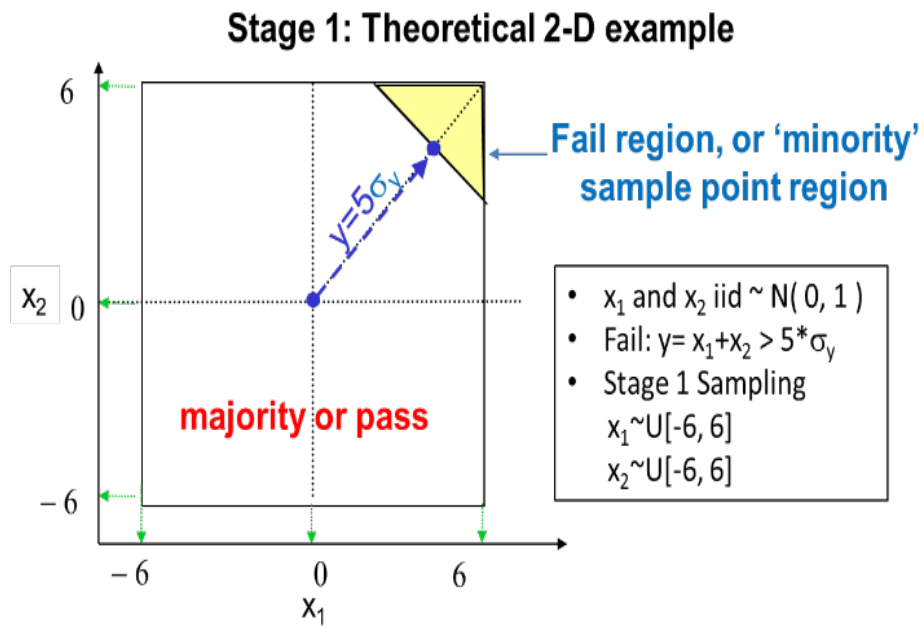


Figure 6.3: Fail Region (shaded). The ratio of the non-shaded to shaded region reflects the ratio of the number of majority sample points (pass points) to minority sample points (fail points) and was used to compute the theoretical ratio in Figure 6.4 [6].

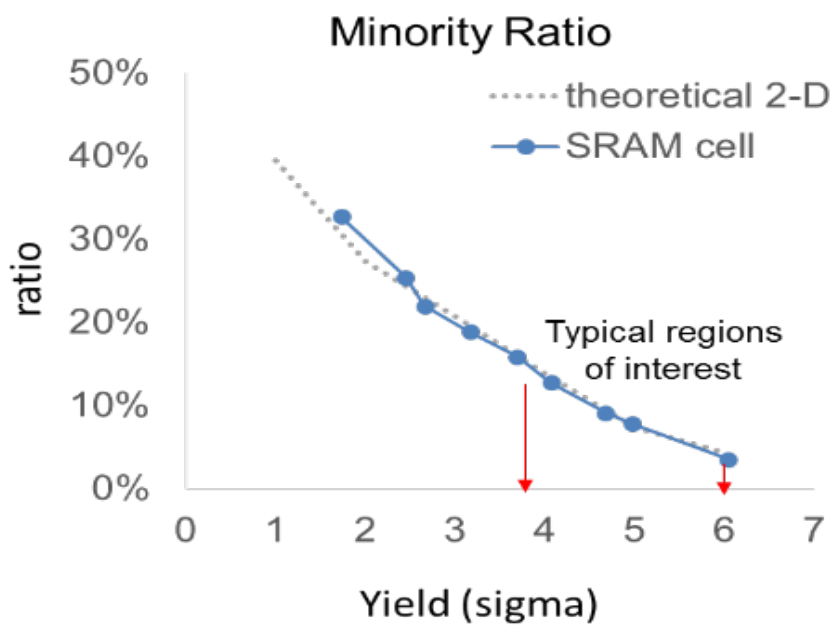


Figure 6.4: The ratio of minority ‘failed’ sample points to the total number of sample points, obtained in stage 1, can drop to few percent for designs with yield values in the typical ranges of interest. The utilized SRAM cell cross-section has a plurality of random variables [6].

# Chapter 7

## Best Balance Ratio Ordered Feature Selection Methodology for SVM applications

Recently, machine learning yield models for Integrated Circuit (IC) have gained widespread prominence in the EDA community, and are very promising in terms of emulating memory design functionality and thereby speeding up circuit simulation based variance reduction methods. A main challenge that arises in this area is class imbalance that occurs naturally due to the high targeted manufacturing yield. Thus, the imbalanced nature of the sampled memory datasets can compromise the model performance.

Support vector machines lends itself as a supervised machine learning technique that is well-known for its generalization capabilities and efficiency. Several works have targetted regularization and data imbalance in the context of SVM. These include implicit methods that jointly address regularization and balancing. In this work, we attain deep insights into the memory classification problem for

modeling rare fail events in the context of importance sampling based yield analysis. We propose a comprehensive and computationally efficient method that addresses the joint considerations of the best combination of relevant features and class balance ratios, which are key for classifier generalization capability. The methodology relies on synthetic minority oversampling techniques to enforce the minority class while probing for the best data balance ratio in conjunction with an iterative  $L_1$ -SVM based approach that qualifies as an approximation to the  $L_0$ -norm regularization for the best feature subset selection. We compare the proposed methodology against standalone  $L_1$ -SVM solutions, unbalanced  $L_0$ -norm approximation as well as an algorithmic data balancing method in the context of yield estimation methodology. The methodology is shown to result in high fidelity classifiers as demonstrated when analyzing the yield of a 14nm Fin-FET SRAM cross-section with a significant speedup for the importance sampling simulations compared to pure circuit simulation based approaches and enhanced accuracy compared to pure  $L_1$ -regularized approaches with clear advantages for probing for the best balance ratio.

## 7.1 Motivation

Accurately estimating rare memory fail probabilities is a challenging task that is relevant to machine learning. Recently, machine learning and big data techniques gained major interest in the area of memory design to develop accurate and fast yield models for memory design yield enhancement. As discussed earlier, we proposed a regularized logistic regression-based framework with ordered feature selection to capture the binary nature of the SRAM fail mechanisms. Indeed, machine learning has become an indispensable mechanism for modeling process

variation effects and accurate yield estimation. However, this area faces many challenges that are recognized as class imbalance problems due to the nature of the rare fail event estimation problem in memory designs [20]. Since traditional machine learning techniques postulate balanced distributions, their performances may not be optimal when using imbalanced datasets [91]. Often, “real” data sets are imbalanced and are dominated by a group of “normal” sample points existing along with a small group of minority “abnormal” sample points [89].

In fact, class imbalance is prevalent in many real-world scenarios, such as anomaly detection, email fraud detection, or computer intrusion detection. It is well known that major differences between the majority and minority group sizes can cause bias in classifiers towards the majority class thereby compromising the accuracy of the classifier [92]. For the memory yield problem, the aspect of rarity is inherent to the SRAM cell fail mechanisms. This naturally results in data imbalance where the “Abnormal” sample points are fail sample points, and misclassifying them as “normal” can impact the accuracy of the final yield estimate. This is especially true for methodologies [90, 1] that rely on the classifier/model outcome to compute the fail probabilities of the importance samples which hinge on and around the fail boundary. Thus, it is crucial to address the issue of data imbalance and properly model the failure boundary for accurate statistical analysis of memory designs.

Support Vector Machine (SVM) is a popular supervised machine learning methodology that is well known for its efficiency, prediction accuracy and robustness [93]. While SVMs have been very effective in dealing with balanced datasets, like other classifiers, SVM could result in “suboptimal results” when dealing with imbalanced datasets.

SVMs are sensitive to class imbalance as a result of different factors. Partici-

ularly, due to the sparseness of the minority class sample. This implies that the low existence of minority examples makes them appear further from the ideal class boundary than the majority ones. In order to lower misclassification, and due to class imbalance, the model may result in a separating hyperplane that is highly skewed towards the minority class [94]. As a result this may cause to low performance on minority sample points.

It has been experimentally demonstrated [95] that the ratio of the majority to minority support vectors becomes also imbalanced in an imbalanced data set. Therefore, a test sample point lying close to the boundary will be surrounded by more majority support vectors. Hence, the model is more likely to classify a boundary point as a majority point particularly for non-separable boundary cases [95, 96]. On the other hand, Akbani et. al [97] stated that the coefficients of the minority support vector may be inherently larger reducing to some extent the effect of the data imbalance on the SVM model.

To alleviate the data imbalance problems for classifiers in general and SVMs in particular [94], two forms of solutions were carved: 1) data handling methods, also known as explicit methods, and 2) Algorithmic methods, also known as implicit methods. Data preprocessing methods focus on balancing the ratio between the minority and majority sample points. Thus, they encompass modifications to the imbalanced data by utilizing resampling methods [6]. Recently, there have been several advances in data handling schemes for machine learning applications [89, 98, 99, 100]. These include random oversampling approaches, random under-sampling approaches, and synthetic sample point generation such as Synthetic Minority Oversampling Technique (SMOTE) to re-enforce the minority class. In [101], the authors propose for enhanced accuracy to perform oversampling over a subset of features. Ensemble learning-based explicit methods were proposed

to solve the imbalance problem in SVMs by employing undersampling methods [102]. This method works by sampling the original majority sample points into multiple smaller subsets of the same size as the minority set, and a group of ensemble models are developed accordingly. The different decisions are then generated by the SVM classifiers and a verdict is computed using methods such as majority voting. However, While some practitioners advocate for perfectly balanced class distributions, others still believe that the natural balance should be used. In fact, the authors in [84][103, 104] propose probing for the best class balance stating that the classifier accuracy hinges on the underlying data.

At the algorithmic level, solutions include adjusting the misclassification costs by penalizing the slack variables using different weights for the different classes so as to reverse the data imbalance [105], thereby rendering the SVM more robust to class imbalance. These include the 'different error cost method' proposed by [106] which is a cost sensitive approach that strengthens the soft margin by adjusting the objective function of the SVM and assigning two misclassification costs to avoid any boundary skewness towards the minority sample points. Other implicit methods include, zSVM which removes bias towards majority class [107] and a two-dimensional surface solution that efficiently solves for a family of variable weights L2-regularized solutions for the imbalanced data set cost functions [22]. We later evaluate the effectiveness of this method in relation to the proposed framework [105].

In general, the family of  $L_p$ -norm regularization has been shown to be critical for classification accuracy particularly in the presence of imbalanced data set [108, 109, 110]. The authors in [108, 111] indicated that for real-world datasets having high skew level in the class distribution, with the class of interest being relatively rare, it is necessary to select features that lead to a higher separabil-

ity between the two classes. Thus, in order to accurately capture the high skew in the class distribution feature selection based methodologies need to be applied. Zheng and Srihari [112] propose feature selection framework that properly represent the positive and negative classes by relying on modified filter method measures. Weston et al. We demonstrate effectiveness of L1-norm and/or L2-norm regularization for SVM accuracy.

We propose for the first time a fast importance sampling methodology for memory yield estimation whose simulation engine employs learning models that probe for the best data balance ratio (BBR) in conjunction with the best subset feature selection. The latter is implemented through a bi-level optimization formulation where the outer loop employs ordered feature selection wrapped around an inner L1-SVM loop for enhanced model capabilities.

Specifically our contributions are as follow:

1. At the core of the methodology lies a data balancing scheme with varying balance ratio to handle the class imbalance problem due to the typically low proportion of minority fails in memory design problems. This allows for identifying the Best Balance Ratio (BBR) among a set of ratios ranging between the natural and fully balanced dataset distribution for enhanced classifier generalization capabilities, thereby not ruling out the naturally occurring class distribution in accordance with literature.
2. We rely on Ordered Feature Selection (OFS) approach similar to [1] as an approximation to the otherwise non-convex  $L_0$ -norm regularization for identifying the relevant SVM features for the respective imbalanced datasets with the different adjusted balance ratios. The approach builds upon a semi-smooth newton coordinate descent  $L_1$ -SVM solution engine [21] to

iteratively identify the relevant features for a given dataset. While the  $L1$ -SVM is often performed as a relaxation to the  $L0$ -norm in terms of augmenting the optimization constraints, we demonstrate enhanced accuracy for our  $L0$ -norm approximation.

3. We study an algorithmic 2 dimensional surface solution methodology (2DWSVM) targeting imbalanced data sets SVM classification [22] that works by adjusting the classifier learning algorithms to better recognize the minority class. The methodology thus identifies the optimal cost function weights for the SVM classes as determined among a derived family of solutions. We rely on this approach as our reference implicit data balancing methodology and critically evaluate its classification and balancing capabilities compared to the proposed explicit approach.
4. We study the yield of state of the art industrial 14nm FinFET SRAM design. We compare our proposed BBR-OFS methodology to the  $L1$ -SVM approach, OFS on the natural distribution and the implicit 2DWSVM approach. Our results demonstrate that BBR-OFS outperformed other methods in terms of accuracy while maintaining significant improvement in terms of the runtime efficiency compared to pure circuit simulation based approach. This is manifested both at the model prediction level and yield estimation capabilities.

## 7.2 SVM Overview

Support Vector Machine (SVM) was introduced by Vapnik [113]. SVM is a deterministic supervised machine learning method for binary classification. The

applications of SVM include face detection, text and hypertext categorization, and bioinformatics [114]. It carries rigorous theoretical advantages and promising computational performance merits in terms of: the ability to be generalized well, and the ability to find global and nonlinear solutions by relying on the "kernel trick" [93]. For the purpose of understanding and explaining the SVM classifier, we denote the training data set consisting of  $n$  sample points  $\{(\mathbf{x}_i, y_i); i = 1, 2, 3 \dots n\}$  where  $\mathbf{x}_i \in \mathbb{R}^d$  is a  $d$ -dimensional input feature vector and  $y_i \in \{-1, 1\}$  is the corresponding output. The objective of the SVM is to construct an optimal separating hyperplane in the input space between the two classes such that all vectors on one side of the hyperplane belong to the same class as illustrated in Figure 7.1 while maintaining a maximal distance to the closest points or vectors. It is worth noting that the solution of the SVM is independent of the number of sample points but dependent on the support vectors which are the input vectors closest to the hyperplane [115].

If the dataset is linearly separable, the SVM solves for the discriminant function of the separating hyperplane that satisfies the inequality  $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) > 0$ , where  $\mathbf{w}$  is the  $d$ -dimensional vector and  $b$  is the constant representing the parameters of the hyperplane;  $y_i (\mathbf{w} \cdot \mathbf{x}_i + b)$  is the functional margin of an individual training point. A large functional margin reflects a good generalizability and is an indicator of a correct and accurate calculation [60].

Hence the objective is to maximize the minimum margin according to (7.1);  $\gamma_i$  is the normalized geometric margin of the  $i$ th sample point  $(\mathbf{x}_i, y_i)$ .

$$\max \min_i \gamma_i = y_i \left( \left( \frac{\mathbf{w}}{\|\mathbf{w}\|} \right) \cdot \mathbf{x}_i + \frac{b}{\|\mathbf{w}\|} \right) \quad (7.1)$$

It follows that maximizing the margin is equivalent to minimizing the weights.

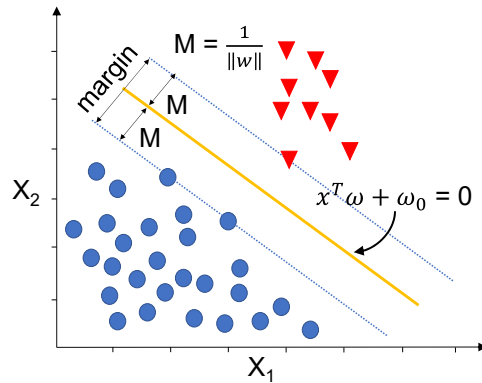


Figure 7.1: 2-D illustration of an SVM separating boundary. Support vectors are sample points closest to the boundary [7].

The linear SVM can thus be reformulated as a convex objective function as illustrated below (7.2). It is also noted that  $\frac{1}{\|\mathbf{w}\|}$  represents the distance from the support vectors to the hyperplane.

$$\text{minimize } \frac{1}{2} \|\mathbf{w}\|^2 \quad (7.2)$$

$$\text{s.t. } y_i (\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 \quad \text{for } i \in 1, 2, \dots, n \quad (7.3)$$

Equation (7.2) can be solved by using traditional quadratic programming. Often, solutions are derived more efficiently from the dual form employing Lagrange multipliers.

When the data is linearly separable, the SVM aims at structural risk minimization and attempts to solve for the optimal hyperplane that maximizes the sum of the distances to the nearest positive and negative training sample points [116]. In cases when data is not linearly separable either non linear SVMs are employed or slack variables are introduced in the optimization model to allow some data points to be within the margin and violate the constraints. Hence, the

SVM optimization problem turns to:

$$\min_{\mathbf{w}, \xi} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i \quad (7.4)$$

$$\text{s.t. } y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i \quad \xi_i \geq 0 \quad \forall i \quad (7.5)$$

$\xi_i$  is the slack variable introduced that relaxes the margin constraint. This gives way to the 2-norm soft margin SVM classifier where  $C$  is the controlling penalty parameter used to attain relative balance between maintaining  $\|\mathbf{w}\|^2$  small and ensuring that most sample points satisfy a good functional margin. Hastie et al. [117] adopted an alternative form that relies on  $\lambda \cong \frac{1}{C}$  and thus (7.5) can be rewritten as:

$$\min_{\mathbf{w}, \xi} \frac{\lambda}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^n \xi_i \quad (7.6)$$

where  $\lambda$  is a regularization parameter and (7.6) offers a balance between model complexity and data fitting. The Lagrange primal function becomes:

$$L_p : \sum_{i=1}^n \xi_i + \frac{\lambda}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^n \alpha_i (1 - y_i f(\mathbf{x}_i)) - \sum_{i=1}^n \gamma_i \xi_i \quad (7.7)$$

where  $\alpha_i \geq 0$  and  $\gamma_i \geq 0$  are the Lagrange multipliers. The system is solved by setting the derivatives to zero along with the Karush–Kuhn–Tucker (KKT) conditions:

$$\frac{\delta L_p}{\delta \mathbf{w}} = 0, \quad \frac{\delta L_p}{\delta \xi} = 0 \quad (7.8)$$

$$\alpha_i (1 - y_i - \xi_i) = 0 \quad (7.9)$$

$$\gamma_i \xi_i = 0 \quad (7.10)$$

SVMs can also handle nonlinear problems by mapping the original input vectors

to a higher dimensional space in order to solve for the separating hyperplane [113]. This can be performed by explicitly relying on feature mapping functions:  $\hat{\mathbf{x}}_i = \phi(\mathbf{x}_i)$ . Alternatively, a kernel function  $K(\mathbf{x}_i, \mathbf{x}_j)$  that satisfies the mercer criterion, i.e.,  $K \in \mathbb{R}^{d \times d}$  is symmetric positive semidefinite, is used to implicitly perform the mapping [113].

### 7.3 Data Balancing Techniques: Implicit

Algorithmic modifications tackling data imbalance work at the heart of the machine learning algorithm.

In the following section, we present 2DWSVM approach as an implicit algorithmic approach that efficiently handles both regularization and data balancing. We present here an overview of other methodologies.

The Different Error Cost(DEC) method targets the SVM algorithm. The SVM algorithm is sensitive to class imbalance since the objective function of the soft margin gives the same misclassification cost for positive and negative class. Hence, the hyperplane would be skewed towards the minority class points. The DEC method comes to play by adjusting the objective function for the classifier and adding two different classification cost one for the positive class  $C^+$  and one for the negative  $C^-$  as shown in the equation below [118].

$$\min\left(\frac{1}{2}w^T w + C^+ \sum_{i|y_i=+1}^l \epsilon_i s.t. y_i(w \cdot \phi(x_i) + b) \geq 1 - \epsilon_i \epsilon_i \geq 0, i = 1, 2 \dots l \right) \quad (7.11)$$

The dual optimization becomes:

$$\max_{\alpha} \quad W(\alpha_i) = \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i \mathbf{x}_j) \quad (7.12)$$

$$\text{s.t.} \quad \sum_{i=1}^l y_i \alpha_i = 0 \quad 0 \leq \alpha_i^+ \leq C^+ \quad 0 \leq \alpha_i^- \leq C^- \quad i \in 1, 2, \dots, l. \quad (7.13)$$

This method reduces the effect of imbalance and the hyperplane won't be skewed towards the minority class. Akbani et al. said that a better results would be obtained if the  $C^+$  and  $C^-$  were equal[119]

The zSVM is another method for implicit data balancing approach. An SVM model is built using the full initial dataset. The bias of the hyperplane is adjusted [120]. Then the decision function is presented as follows:

$$f(x) = \text{sign}\left(\sum_{i=1}^l y_i \alpha_i K(x_i, x) + b\right) = \text{sign}\left(\sum_{i=1}^{l_1} y_i \alpha_i^+ K(x_i, x) + \sum_{j=1}^{l_2} y_j \alpha_j^- K(x_j, x) + b\right) \quad (7.14)$$

In the equation presented above,  $\alpha_i^+$  is the positive support vector coefficient and  $l_1$  is the number of positive samples. On the other hand,  $\alpha_i^-$  refers to the negative support vector coefficient and  $l_2$  is the number of negative samples. When performing this method the value is increased by being multiplied by a small positive value  $z$  and hence lowering the bias.

Kernel modification methods are techniques that perform implicit data balancing for SVMs by performing adjustment to the kernel method. This is done by the kernel alignment target method. It evaluates the agreement between the kernel and classification task which enhances the data imbalance a lot [121].

## 7.4 2D WSVM

In classical SVM the data points are treated equally, however in certain scenarios this may not be optimal and applying certain weights to the data points can improve the accuracy of the model. Wahba et al. [105] introduced the concept of weighted SVM (WSVM) where the cost function employs a weight parameter  $\pi_i$  to assign different weights to the different classes. The formulation in (7.18) can be rewritten as (7.15), and the the Lagrange primal for the hinge loss becomes (7.16).

$$\min_{f \in H} \sum_{i=1}^n \pi_i L(y_i, f(\mathbf{x}_i)) + \lambda J(f) \quad (7.15)$$

$$L_p = \sum_{i=1}^n \pi_i \xi_i + \frac{\lambda}{2} \|\omega\|^2 + \sum_{i=1}^n \alpha_i (1 - y_i f(\mathbf{x}_i)) - \sum_{i=1}^n \gamma_i \xi_i \quad (7.16)$$

where weight  $\pi_i = \pi$  for the majority class points ( $y_i = -1$ ) and  $\pi_i = 1 - \pi$  for the minority class points ( $y_i = 1$ ).  $0 \leq \alpha_i \leq \pi$  and  $\gamma_i \geq 0$  are the Lagrange multipliers. Similar to section 7.5.1, the algorithm hinges on the Elbow partition, the left partition and the right partition.  $\alpha = 0$  implies a point in the Right disjoint set, and  $\alpha = \pi$  implies a point in the Left disjoint set. For a fixed  $\pi$  value, the standard  $\lambda$ -path algorithm [122] can be employed. A  $\pi$ -path algorithm was also developed to efficiently determine  $\pi$  solutions for fixed  $\lambda$  values [123].

The authors in [22] demonstrate that the solution surface is jointly piecewise linear in  $(\lambda, \pi)$  pair so long that no event happens. An event pertains to a non-support vector becoming a support vector or vice versa. Thus, they propose a two-dimensional solution surface that efficiently computes the entire solution surface in the 2-D  $(\lambda, \pi)$  space. As  $\lambda$  and  $\pi$  change the sets change and the solutions are continuous. The methodology relies on recursively identifying subregions or polygons,  $Sur f^l$  defined by their respective vertices  $v_r^l = (\lambda_r, \pi_r), r = 1, 2 \dots n_v$

where  $n_v$  signifies the number of vertices as determined by the constraints that prevent an event from happening. The method [22] then exploits the piecewise linearity of the solution to extend the solution until the whole solution surface space is covered. The solution is finer in regions where the solution is complex and coarser otherwise. In addition to the computational efficiency, this offers an advantage over a uniform grid-based solution surface.

## 7.5 BBR-OFS Framework for Yield Estimation Methodology

In this section, we introduce the proposed framework for efficient statistical analysis and modeling for importance sampling based rare event estimation. Figure 7.2 presents the flow diagram of the overall methodology.

The methodology relies on importance Sampling as its fast Monte Carlo yield estimation engine [15]. To speedup the simulations, at the core of the methodology lies a classification approach that incorporates data balancing along with feature selection for enhanced accuracy in the event of imbalanced data sets. Specifically, the proposed methodology relies on scouting for the best balance ratios in a bi-level optimization approximation to  $L_0$ -norm regularization to enhance prediction accuracy and reduce simulation cost. First, we introduce the  $L_0$ -norm approximation where we employ ordered feature selection as a wrapper around a fast-newton method based  $L_1$ -SVM method to guide the model building phase. Then, we implement a general framework that identifies the best class balancing ratio in the context of the SVM  $L_0$ -norm approximation.

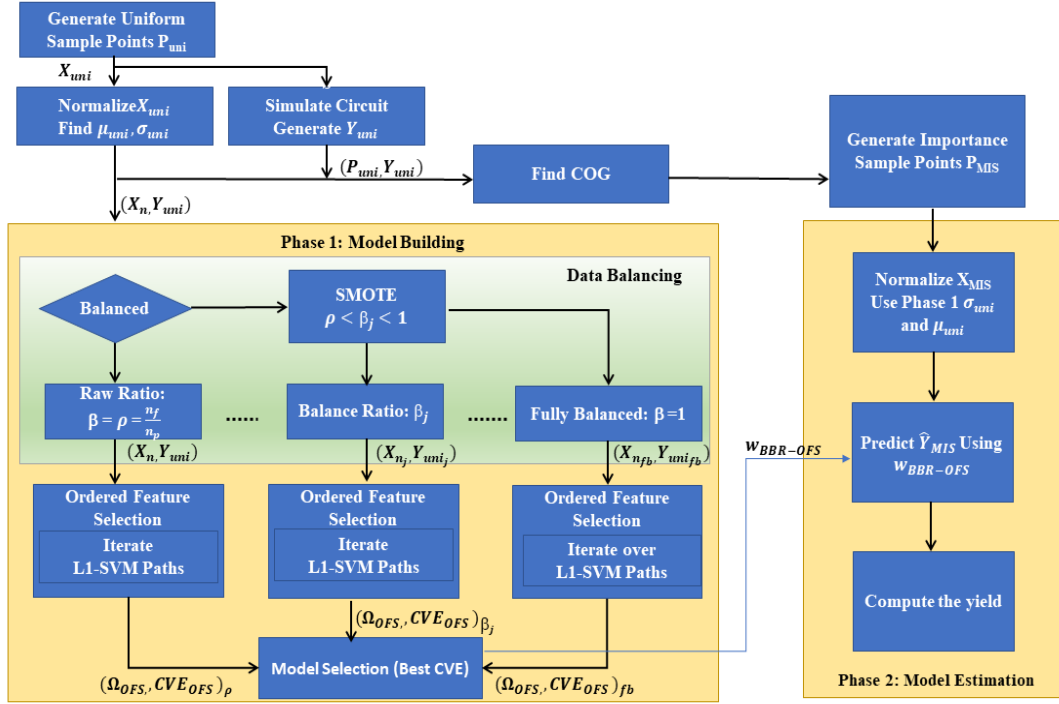


Figure 7.2: Methodology Flow Diagram for the proposed framework.

### 7.5.1 Ordered Feature Selection

Feature selection is critical for "generalization" performance, challenging the curse of dimensionality, and even lowering the time needed to train a model [124]. Particularly, for SVMs, irrelevant features and noise may jeopardize the accuracy, convergence and efficiency of the classifier [109]. Hence, finding minimal subset of features that gives maximal generalization capability using feature selection is desired. Traditionally, algorithms with feature selection address two rivaling objectives: maximizing the goodness of fit and minimizing the number of variables [124]. The latter is very similar to regularization objectives especially those targeting shrinkage. Feature selection methods lie in three main categories: the wrapper, the filter, and embedded methods [124]. The wrapper method exhaustively calls the training model to introduce at each step the next

best feature candidate. It typically relies on cross-validation for best candidate selection. The filter method applies a preprocessing step to rank variables based on a statistical score independent of the training algorithm. This presents a challenge as it fails to take into account interdependencies between the features. For imbalanced datasets, wrapper and filter methods are the most commonly used methods. Methods such as the wrapper method, however, can be expensive. Alternatively, embedded methods where feature selection is embedded as part of the training algorithm can be used. These methods are typically more accurate than filter methods and are less prone to overfitting. Embedded methods that rely on  $l_p$ -norm regularization,  $p \in \{1, 2\}$  have been traditionally employed. Thus, the SVM problem in (7.6) along with its constraints (7.4) can be reformulated in to a "Loss + Penalty criterion" of the form (7.17) which can be generalized to (7.18) [125, 117].

$$\min_{w_0, \mathbf{w}} \sum_{i=1}^n [1 - y_i(w_0 + \mathbf{w} \cdot \mathbf{x}_i)]_+ + \frac{\lambda}{2} \|\mathbf{w}\|_p \quad (7.17)$$

$$\min_{f \in H} \sum_{i=1}^n L(y_i, f(\mathbf{x}_i)) + \lambda J(f) \quad (7.18)$$

$L(y_i, f(x_i))$  is a loss function such as the Hinge loss or the Huber loss function [122].  $f(x)$  is an arbitrary function in some Hilbert space  $H_K$ , and  $J(f)$  is a norm in a Reproducing Kernel Hilbert Space of functions to measure the "roughness" of  $f$  in  $H_k$ . Works such as [126, 127] cited that  $L_1$ -norm was sufficient to reduce a decent number of variable weights to zero. In this context, Least angle regression shrinkage (LARS)-SVM and Least Absolute Shrinkage and Selection Operator (LASSO)-SVM solvers have been proposed [128, 21]. If the model requires a large number of features for better performance, [128] recommend using  $L_2$  regularizers in conjunction with  $L_1$ . Others such as Weston et al., [109] and the authors in

[129], however, proposed  $L_0$ -norm approximations for improved feature selection and enhanced modeling capability. The authors in [129] stated that  $L_0$ -norm can be enforced by adding a cardinality constraint that restricts the number of the model coefficients, [129]. This, however, adds complexity to the optimization problem due to the non-convexity of the constraint. Hence, they proposed to replace it by a weaker non-convex constraint involving the  $L_1$ -norm and  $L_2$ -norm as illustrated in Eqn. (7.19) that relies on the Cauchy-Schwartz inequality and can be further be relaxed into weaker convex constraints [129].

$$\text{Card}(w) \leq r \iff \|w\|_1 \leq \sqrt{r}\|w\|_2 \quad (7.19)$$

Weston et al. [109] proposed an approximate iterative approach that is reminiscent of backward elimination and solves for the  $L_1$ - or  $L_2$ -norm solutions at each iteration while re-scaling the variables by the weight vector. The authors demonstrate that the blend of the  $L_0$ -norm approximation with the underlying regularization counterpart offers significant improvement for generalization capability of the classifier.

Herein, we propose to enforce the best finite set of features  $\Omega$  and accordingly the  $L_0$ -norm approximation by jointly optimizing the following problem.

$$\min_{\Omega \in F} \min_w \sum_{i=1}^n L(y_i, f(x_i)) + \lambda \|w\|_1 \quad (7.20)$$

where  $L(y, f(x))$  is the huberized hinge loss function described in Eqn. (7.23) [130].  $f(x) = wx + b$ , and  $y \in \{-1, 1\}$  and  $h(x) = \text{sign}(f(x))$  is the decision rule used to determine the classification criterion.  $F$  is the feature set, and  $\Omega$  represents a feature subset in  $F$ . Rather than exhaustively searching for the best subset  $\Omega$ , we base the outer optimization on an ordered feature selection (OFS)

approach similar to [1] as illustrated in Algorithm 5. The methodology initially starts by solving for an  $L_1$ -SVM path,  $\omega_F(\lambda)$ , over the whole feature set  $F$  [131]. Since the regularization parameter  $\lambda$  affects the SVM model parameters tuning over the solution path, we pick the solution  $\omega_F(\lambda_F^{min})$  corresponding to the lowest cross validation error. As such, we define an ordered index set,  $Idx_F$  that represents the features in  $F$  sorted in the decreasing order based on their respective model parameters,  $\omega_F(\lambda_F^{min})$ . This ordered index qualifies as a combined embedded filter rank metric. In fact, it has been known that although shrinkage methods such as LARS and LASSO are not "full-fledged" wrapper methods, these methods are often perceived as a semi-wrapper methods with the ability to order features based on their significance [128]. The algorithm then proceeds to incrementally build the best feature set,  $\Omega_{OFS}$  by adding at each iteration the next feature according to  $Idx_F$ , and solving for the  $L_1$ -SVM path in a forward selection manner. At each iteration, the best solution along the path is chosen by relying on k-fold cross-validation. The best ordered Feature subset and model for our data is determined as the ordered subset with lowest cross-validation error among all subsets. The ordered Feature selection flow is then applied to data set subject to different balancing ratios as will be discussed in the following subsection to determine the best balance ratio and feature subset combination. k-fold cross-validation offers an excellent estimate of the SVM generalization error and thus model selection. To perform k-fold cross-validation, the data is split into  $K$  partitions as illustrated in Figure 7.3. The model error is collected over  $K$  iterations. For each iteration one partition is used for testing and the remaining  $K - 1$  partitions are used for training. The CVE error, can thus be computed as the average Mean Squared error,  $MSE$ , of the test partitions across the  $K$

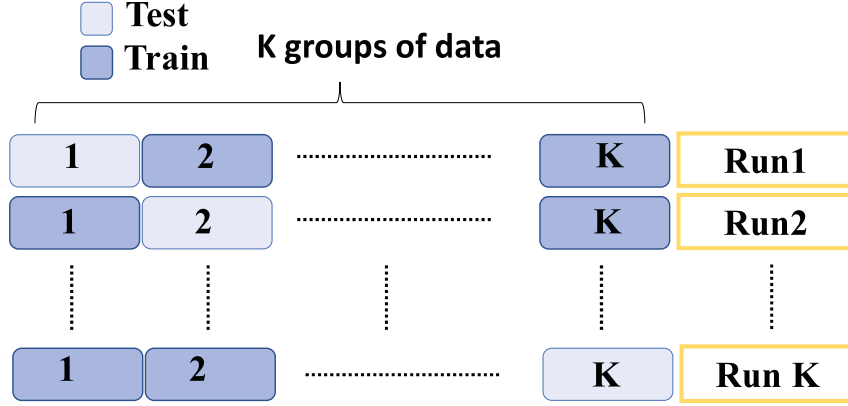


Figure 7.3: K-fold cross Validation [4].

iterations as follows [132]

$$MSE_k(\lambda) = \frac{1}{n/K} \sum_{i=1}^{n/K} (y_i - x_i \hat{w}^{-k}(\lambda))^2 \quad (7.21)$$

$$CVE(\lambda) = \frac{1}{K} \sum_{i=1}^K MSE_k(\lambda) \quad (7.22)$$

where  $n$  represents the number of sample points in the dataset. For our purposes, to obtain the  $L_1$ -SVM solution path, we adopt a LASSO-based sparse SVM approach that relies on a semi-smooth newton coordinate algorithm [131, 21] to efficiently solve a huberized hinge loss based regularization. In fact, it has been shown that the complexity relies on identifying the proper loss function [133]. While Eqn. 7.17 can be mapped in terms of the hinge Loss function  $L(y_i f(x_i)) = [1 - y_i f(x_i)]_+$ , it has been shown that the Huberized Hinge loss function in (7.23) which is almost quadratic offers differentiability that reduces the computational cost and at the same time maintains linearity similar to the

hinge loss for negative margins to prevent outliers from affecting the model.

$$L_s^{HH}(y_i f(x_i)) = \begin{cases} 0, & y_i f(x_i) > 1 \\ \frac{(1 - y_i f(x_i))^2}{2\delta}, & 1 - \delta < y_i f(x_i) \leq 1 \\ 1 - y_i f(x_i) - \frac{\delta}{2}, & y_i f(x_i) \leq 1 - \delta \end{cases} \quad (7.23)$$

The solution path  $w(\lambda)$  is thus piece-wise linear in  $\lambda$ , and the piece-wise linearity "hinges" on the following sample point set partitions associated with  $L^{HH}$  [134] that stem from the definition of the loss function (7.23), i.e., sets 1, 2, and 3 below. The Elbow partition thus represents points that are on the margin (non-bound support vectors), the left partition holds sample points that lie inside the margin (bound support vectors) and the right partition contains the points that lie outside the margin. The solution also hinges on the active feature set  $\nu \in \Omega$  that is also function of  $\lambda$ .

1. Right :  $R = \{i \mid y_i f(x_i) > 1\}$
2. Elbow:  $E = \{i \mid 1 - \delta \leq y_i f(x_i) \leq 1\}$
3. Left:  $L = \{i \mid y_i f(x_i) \leq 1 - \delta\}$
4. Active features set:  $\nu \mid \{j : w_j \neq 0\}$

Hence, the solution path according to the KKT conditions starts by adopting a large  $\lambda$  value and proceeds to decrease  $\lambda$  gradually until we reach an event pertaining to a training point hitting  $E$ , set of points qualifying as support vectors, or a feature leaving  $\nu$ , the set of active features with nonzero coefficients. Such events incur a "kink" in the solution and the derivatives of the parameters with respect to  $\lambda$  will change. In between events, the solution for  $\omega$  is linear in  $\lambda$ .

---

**Algorithm 5** Ordered Feature Selection [7]

---

**Input**

$X$ : Input Features

$Y$ : Output Data

**Output**

$\Omega_{OFS}$ : OFS Resultant ranked feature subset  $CVE_{OFS}$ : Minimum cross validation error

- 1: **function** L1-SVM( $X, Y$ )
- 2:     Find SVM solution path  $\omega(\lambda)$  over  $\lambda \in [0, \lambda_{large})$
- 3:     Find  $\lambda^{min} \mid CVE^{min} = \min(CVE(\lambda^k)) \forall \lambda^k$
- 4:     **return**  $\omega(\lambda^{min}), CVE^{min}$
- 5: **end function**
- 6: **function** OFS( $(X, Y)$ )

**Initialization**

- 7:      $CVE_{OFS} = maxNum$ ;
- 8:      $\Omega_{OFS} = \{\}$
- 9:     Find Solution over all Features  $X$
- 10:     $\omega_F(\lambda_F^{min}) = \text{L1-SVM}(X, Y)$
- 11:    Sort the  $\{\omega_F(\lambda_F^{min}) \neq 0\}$  model parameters
- 12:     $Idx_F, \mid \omega_{sort}(i) = \omega_F(\lambda_F^{min}, Idx_F(i)) \forall \omega_F \neq 0$
- 13:    **while**  $j < \min(size(Idx_F, maxNumFeat)$  **do**
- 14:        $\Omega_j \leftarrow X(Idx_F(1 : j))$
- 15:        $[\lambda_j^{min}, CVE_j^{min}] = \text{L1-SVM}(\Omega_j, Y)$
- 16:       **if**  $CVE_j^{min} < CVE_{OFS}$  **then**
- 17:           $CVE_{OFS} = CVE_j^{min}$
- 18:           $\Omega_{OFS} = \Omega_j$
- 19:       **end if**
- 20:    **end while**
- 21: **end function**

As discussed earlier, imbalanced classification is one of the most challenging problems encountered in the field of machine learning and has been prevalent in several fields. A dataset is defined as imbalanced when the two class are not equally represented [79]. Imbalanced datasets have been shown to compromise the predictive capabilities of the model. For SVM class prediction, it has been demonstrated as discussed earlier that the boundary may be too skewed towards the minority sample points in the event of imbalanced data sets and a point close to minority may be classified as majority. Hence, the weakness of the soft margin can be a cause for the performance loss [97]. In an extreme scenario for imbalanced data sets, smaller  $C = \frac{1}{\lambda}$  values, i.e., smaller penalty on the slack variable in (7.6), may result in increasing the margin and causing the SVM to assign all points as majority points. This results in sacrificing the minority points as a consequence of relaxing the margin with the only error to be accounted for being the few minority sample points that are missed. Hence, SVM may in some scenarios not generalize well when addressing imbalanced datasets. To handle the imbalance, researchers have proposed to rely on resampling the data sets (oversampling or undersampling) to reduce the rarity of minority sample points prior to building the classifier. They also recommend to employ in conjunction  $L_p$ -norm regularization along with appropriate evaluation metrics such as cross validation [135, 136, 137]. In [103], the authors show that the ideal class distribution is not necessarily the fully balanced distribution and as such attempt to find a class distribution that improves the generalization capability of the classifier. The authors further demonstrate that the results can be enhanced should intelligent resampling techniques be used. In fact, Weiss and Provost [138] showed that the ideal class distribution depends a lot on the learning algorithm used for enhancing the modeling capability. Synthetic minority oversampling Technique

(SMOTE) is a popular data balancing method shown to improve the model predictive capability in SVMs [79]. In particular, it has been shown to outperform other sampling methods and the quality improvement was measured by different model quality indicators [6]. In this work, we employ SMOTE as the resampling technique in our pursuit for the best data balance ratio based OFS SVM model. We also compare our approach with an algorithmic-based balancing approach for SVMs that is discussed in Section 7.3.

The dual optimization in (7.20) along with the search for the best data balancing ratio comprise the core circuit modeling engine for the proposed yield estimation framework of Figure 7.2. Together, they approximate a multi-level optimization that can best be described in (7.24).

$$\min_{\beta_j} \min_{\Omega \in F} \min_w \sum_{i=1}^n L(y_i, f(x_i)) + \lambda \|w\|_1 \quad (7.24)$$

where  $\beta_j$  represents a varying data balancing ratio as presented in pseudo code of algorithm 6 that summarizes the flow of the proposed modeling and yield estimation framework. The importance sampling yield estimation methodology encompasses two phases: a first uniform sampling phase to determine the center of gravity of fails which comprises the shift center for the distribution in the second importance sampling phase. We leverage the circuit simulations performed in the first phase [1] to build a model that can be employed to eliminate the need for circuit simulations in the following phase. For Phase 1, referred to hereon as the model building phase, the original data is preprocessed and the dataset is analyzed to determine the level of imbalance  $\rho = \frac{n_f}{n_p}$ , where  $n_f$  is the number of failing (minority) sample points and  $n_p$  is the number of passing (majority) sample points. An assorted set of minority oversampling ratios,  $\rho \leq \beta_j \leq 1$ ,

spanning the range between the natural raw ratio,  $\rho$  and the fully balanced one is determined. We, then, rely on SMOTE to generate synthetic sample points corresponding to these ratios. For each resulting dataset with a specific balance ratio, OFS is employed as discussed in Section 7.4.1 resulting in multiple models for the different balance ratios. Among these models, the model corresponding to the lowest cross-validation error is adopted. This model thus comprises the BBR-OFS model that employs the critical set of features and minority oversampling ratio that enhances the model fidelity in order to enable efficient and accurate yield estimation. In Phase 2, importance sample points generated using the center of gravity of fails as determined in Phase 1 are evaluated and their respective performance metric is derived using the BBR-OFS model. The yield estimate is then derived according to (7.25). Maintaining high model fidelity is important especially when using the framework as a comparator among several high-sigma yield design points which can be otherwise a challenging requirement.

$$\hat{P}_f = \frac{1}{M} \sum_{i=1}^M I(x) \frac{pdf(x_i)}{pdf_{IS}(x_i)} \quad (7.25)$$

---

**Algorithm 6** Yield Estimation Methodology [7]

---

**Input** $P_{uni}$ : Process variation parameters for uniform sampling $F_{uni}$ : Performance metric vector**Output** $\sigma_{IS}$ : Importance sampling yield estimate based on evaluating response for  $P_{MIS}$ 

---

*Phase 1 – BBR-OFS Model Building*

---

- 1: Preprocess Data: Generate polynomial and interaction terms using Hermite functions  $g_i$  [64]
- 2:  $X_n = \frac{X_{uni} - \mu_{X_{uni}}}{\sigma_{X_{uni}}}$
- 3:  $Y_{uni} = (F_{uni} > f_0)$  //  $f_0$  represents fail criterion
- 4:  $X_{n_{min}} = \{X_n | (Y_{uni} == 1)\}$  // minority set
- 5: Let  $\rho = \frac{n_f}{n_p} = \frac{size(X_{n_{min}})}{size(X_n) - size(X_{n_{min}})}$  // raw ratio
- 6:  $\beta \in [\rho : 0.1 : 1]$  //  $\beta=1$  fully balanced ratio
- 7: **for**  $j=1 : size(\beta)$  **do**
- 8:  $X_{n_j} = \{X_n \cup \text{SMOTE}(X_{n_{min}}, \frac{\beta_j}{\rho} - 1, k = 5)\}$
- 9:  $Y_{uni_j} = \{Y_{uni} \cup \text{Ones}((\frac{\beta_j}{\rho} - 1) * n_f)\}$
- 10:  $[\Omega_{OFS}, CVE_{OFS}]_{\beta_j} = \text{OFS}(X_{n_j}, Y_{uni_j})$
- 11: **end for**
- 12:  $\beta_{bbr} | CVE_{OFS}_{\beta_{bbr}} = \min(CVE_{OFS}_{\beta_j})$
- 13:  $\Omega_{BBR-OFS} = \Omega_{OFS}_{\beta_{bbr}}$
- 14:  $\omega_{BBR-OFS} = \begin{cases} \omega_{OFS}_{\beta_{bbr}}, & \forall X_n \in \Omega_{BBR-OFS} \\ 0, & \forall X_n \notin \Omega_{BBR-OFS} \end{cases}$

---

*Phase 2 – Model Prediction and Yield Estimation*

---

- 15:  $\mu_{COG} = \text{mean}(\frac{P_{uni} * Y_{uni}}{n_f})$  //Center of Gravity of Fails
  - 16:  $P_{MIS} \leftarrow$  Generate process variation importance sample points around  $\mu_{COG}$
  - 17:  $X_{MIS} = \{g_i(P_{MIS})\}$
  - 18:  $X_{n_{MIS}} = \frac{X_{MIS} - \mu_{X_{uni}}}{\sigma_{X_{uni}}}$
  - 19:  $Y_{MIS} \leftarrow \text{EvalSVM}(X_{n_{MIS}}, \omega_{BBR-OFS})$
  - 20: Calculate  $\sigma_{IS}$  Yield according to (7.25)
-

# Chapter 8

## Results

In this section, we provide an overview of the experimental setup of the circuits under study which we use to test the proposed methodologies. We also report the results obtained from the experiments and demonstrate the effectiveness of the methodologies. We start by providing a brief introduction to FinFETs, FinFET SRAM design and design metrics that will be utilized.

### 8.1 Experimental Setup

#### 8.1.1 FinFET DEVICES

The FinFET device was structured to address technology scaling challenges that prevail beyond the sub 20nm node in planar technology. The thin body of the FinFET device represses a lot of the short channel effects, and this allows the body channel to be lightly doped. This in turn helps reduce the threshold voltage variation. It also lowers the traverse electric field and thus reduces the impurity scattering and carrier mobility. It is also characterized by a steep sub threshold slope, lower leakages as well as enhanced performance compared to planar devices.

Moreover, FinFET devices enjoy lower capacitance which helps lower the memory bit-line loading and enhance the SRAM functionality. Thus, FinFET device have better outlook and great potential in the context of SRAM cell design. There are two types of FinFET devices:

- Double-gate devices: the two gates are biased together to switch the FinFET on/off.
- Back-gate devices: the two gates are biased independently. Front gate is employed to switch the FinFET on/off and the back-gate is used to change the threshold voltage of the device allowing for further tuning.

### **8.1.2 FinFET SOI SRAM**

The FinFET design allows for enhanced scalability which makes it appealing for memory designs. A 6-T SRAM cell is shown in Figure 8.1. The conventional 6T SRAM cell metrics for evaluating the functionality of the SRAM cell and design trade-offs are applicable for FinFET SRAM design as discussed below.

#### **SRAM design requirements and functionality metrics**

As mentioned earlier, the SRAM cell needs to be properly designed to meet area/power tradeoffs. The leakage power can be improved by increasing the threshold voltage, however this can affect the design performance. Often designers target to operate at  $V_{min}$ , the minimum supply voltage that meets the functionality requirements, in order to save power. Dual supply designs, help relax the requirements on  $V_{min}$  [139].

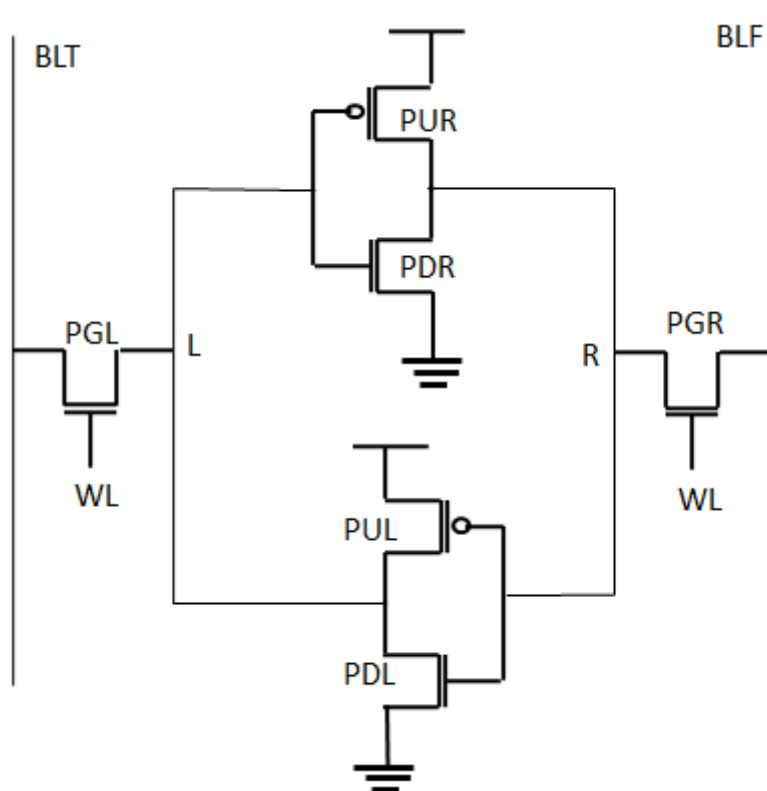


Figure 8.1: Schematic showing a 6-T SRAM

## SRAM Metrics

Hence, there is a need to find the minimum supply voltage for proper SRAM cell functionality and yield. Designers study the cell yield under different stability, writeability and readability tests. The static noise margins were first relied upon in order to study the cell stability. However, particularly for PD/SOI technologies, device-to-device fluctuations can cause cell instability during the basic operations read/write even if static margin is large [139]. Therefore, preferred SRAM performance metrics are the dynamic ‘read stability’ and ‘writeability’. These criteria are more representative since they are the result of transient simulations. Not only do they show the SOI body effects, but they also take into consideration the effect of cell placement and peripheral circuitry as will be discussed next. In the following discussion, we will assume that the SRAM cell is storing ‘0’ on node L and ‘1’ on node R.

- **Read Stability:** When one is reading ‘0’, the access and pull down transistors, PG and PD respectively behave as a voltage divider circuit between the precharged bitline, BLT, and node L. This tends to cause noise at node L. If the process variations are large, the resulting noise may be large due to the cell being weak as a result of the process variations. In such case, it may cause the contents of the cell to flip. This is referred to as a destructive read. In order to have a stable cell the maximum noise at node L must meet a certain conservative threshold around  $1/3 \cdot V_{dd}$ . Typically, during the statistical analysis simulation we count the cells that do not meet the threshold as failing cells. If a threshold is not defined we simply count the flipping cell as a failing cell. Statistical analysis for the read stability of the cell require checking if the cell violates this criteria at a given set of

sample points in the process parameters space [140]. This tends to account for those samples that correspond to the cell flipping.

- **Writeability:** A cell is defined as writeable if its contents can be flipped. A measure of the writeability is the time it takes a cell to write a ‘0’ to a node that was previously ‘1’. This is done by writing a ‘0’ on node R for the cell of Fig 10. Due to the structure of the cell having cross coupled inverters, an adopted metric by the designer community involves measuring the write ‘0’ delay from the time the wordline, WL, gets activated to the time node L charges up; i.e. , write ‘0’ on R is achieved only if only if node ‘L’ gets charged up. Equation 8.2 illustrates the acceptance criteria for write functionality. When this criteria is violated we consider that the cell has undergone a write failure, and this can happen due to write delay (Time to Write) being more than the delay criteria or the current desired operating clock frequency, or the cell not being capable of flipping its contents at all independent of the frequency constraints due to severe process variations. Larger values of k 8.2 are required to ensure that a cell is not mistakenly stuck at its bi-stable state point for a cell that has successfully flipped.

$$T_{write} = (t_2 - t_1) < \tau \quad (8.1)$$

where  $\tau$ : write delay criteria

$$t_1 : VWL(t_1) = \frac{VDD}{2}; t_2 : VL(t_2) = k * VDD, \frac{1}{2} < k < 1 \quad (8.2)$$

## 8.2 Write-Assist Circuitry for FinFET SRAM Design

In planar CMOS technology, designers rely on pulldown-to-passgate (PD/PG) and pullup-to-passgate (PU/PG) device ratios to optimize the SRAM cell  $\beta$  and  $\gamma$  ratios for proper functionality. This allows for robust cell designs while tackling the central design considerations which include: compactness, functionality and performance. Nevertheless, FinFET SRAM designs have several challenges in terms of sustaining proper device sizing that arise as a result of quantization. A minimum-sized FinFET SRAM cell has the PU, PD, and PG devices set to a single fin each. Accordingly, the  $\beta$  and  $\gamma$  ratios are set to one. This negatively affects the functionality at low voltage operation, and necessitates having robust write-assist circuitry and selective boosting techniques for enhanced cell read and write [9]. The authors in [9] propose selective boosting techniques along with write assist circuitry for improved FinFET SRAM design. The circuitry can be used to selectively boost voltages ( $V_{ddv}$ ) along a specific path or set of paths including the write drivers, wordline and the cell as illustrated in Figure 8.2. This improves the cell yield while preserving operation at low voltage level  $V_{dd}$ .

Figure 8.3 illustrates the boosting circuitry schematic which includes an nFET and apFET device in parallel. When the gate switches, it couples the  $V_{ddv}$  to a value above  $V_{dd}$ . This boosts the virtual supply node feeding the selected memory paths by more than 100mV. Whereas the other parts of the circuit operate at  $V_{dd}$ .

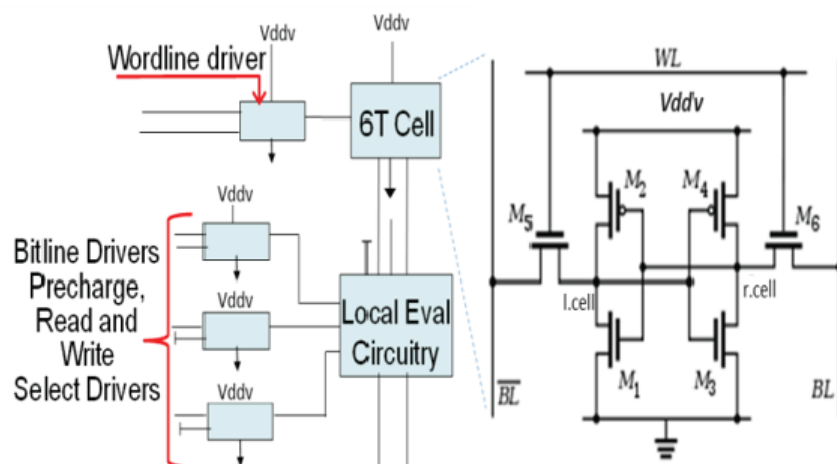
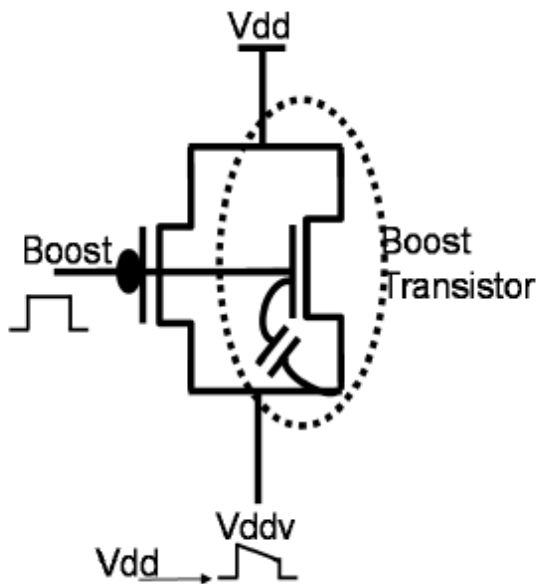


Figure 8.2: Sketch of the SRAM cell cross-section including wordline, bitline and write drivers [8]



### FinFET Boost Transistor

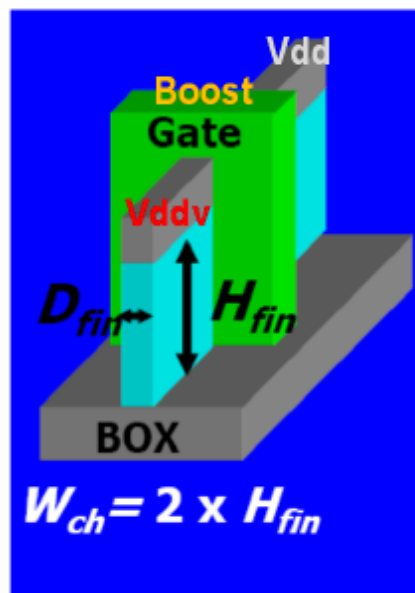


Figure 8.3: Capacitive Coupling between Gate and Source boosts the source Voltage ( $V_{ddv}$ ) when Boost switches "High" [8].

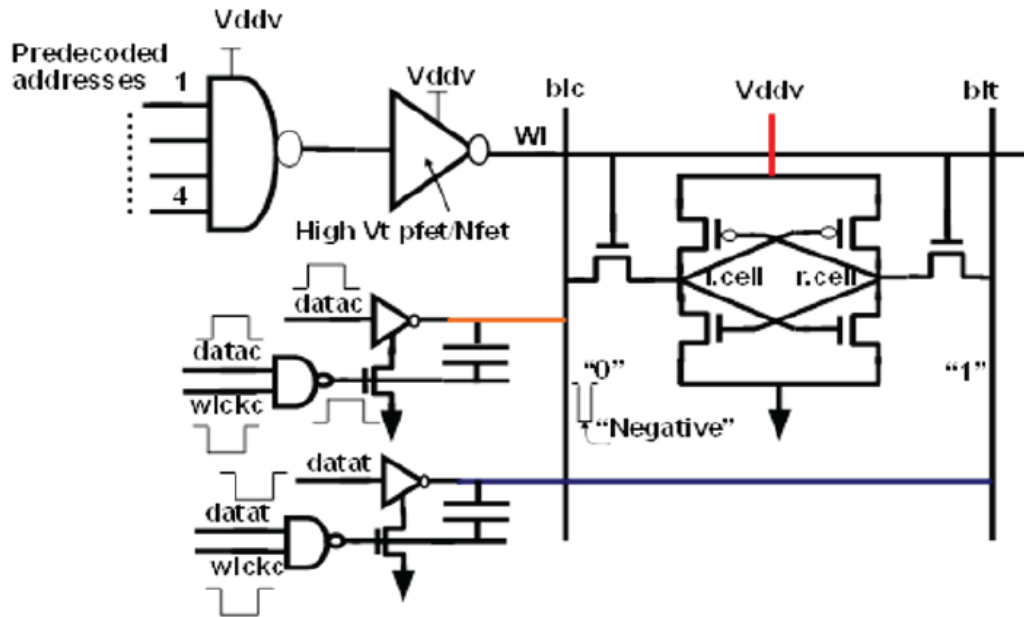


Figure 8.4: Negative Boost Write Assist is applied to the bitlines during write operation. Cell, write driver, and wordline receive boosted supply voltage [9].

### 8.2.1 Experiments

We test the effectiveness of the methodologies proposed by analyzing the 14nm FinFET SRAM designs presented in table 8.1. For accurate analysis and for purposes of model to hardware validation, our simulations include the peripheral logic along with the SRAM cell as illustrated in Figure 8.2. Thus, our cross-section includes the write driver, bitline driver, wordline driver and local evaluation circuitry. Since we are evaluating an SRAM write assist circuit, we mainly focus on writeability in terms of the ability to flip the contents of the SRAM cell. Hence, we analyze the SRAM cell operation in terms of its dynamic write margin. In particular, regarding the cell writability, we define it as the ability to write a ‘0’ or ‘1’ onto the cell internal nodes. In the event of process variations that can have implication on the cell functionality, It is possible due to process variations, that the cell pullup device becomes strong (PUL), and the

pass gate becomes weak (PGL), thereby leading to a writeability fail. This only makes it harder to write a ‘0’ to the SRAM. In simulations, we perform process variations to the SRAM cell transistors and critical transistors of the local evaluation circuitry. For every device, we lump the different sources of variability in terms of metal gate granularity, line edge roughness, fin height, and random dopant functions into an effective  $\sigma_{V_t}$  device threshold voltage is then subject to independent random process variations  $\delta_{V_t} \sim N(0, \sigma_{V_t}^2)$  result in 45 total features. Our purpose is to ensure Vmin improvement due to write assist circuitry. Hence, we employ the proposed methodology to analyse the yield of the Selective Boost design (standard) for Vdd=[0.40V-0.43V] and the Selective Boost with negative bitline Boost write assist WA (Boosted) design yield for Vdd=[0.35-0.40V].

Table 8.1: Summary of the designs used. (a) The Standard design refers to the selective boost design with no write assist. (b) The Boosted design refers to the presence of negative bitline boost for write assist.

Design	Write Assist	Referred As
Selective Boost Only	None	Standard
Selective Boost with Write Assist	Negative Bitline Boost	Boosted

We compare the results to those obtained from a pure circuit simulation based importance sampling approach. We also compare Vmin to hardware data obtained from a 72Kb SRAM arrays (Figure 8.5) that were allocated in columns consisting of 16 cells/bitlines [9]. Hardware measurements show an operating voltage of around 0.35V for the Boosted write assist design. These results were also validated by the statistical yield analysis methodology [141].

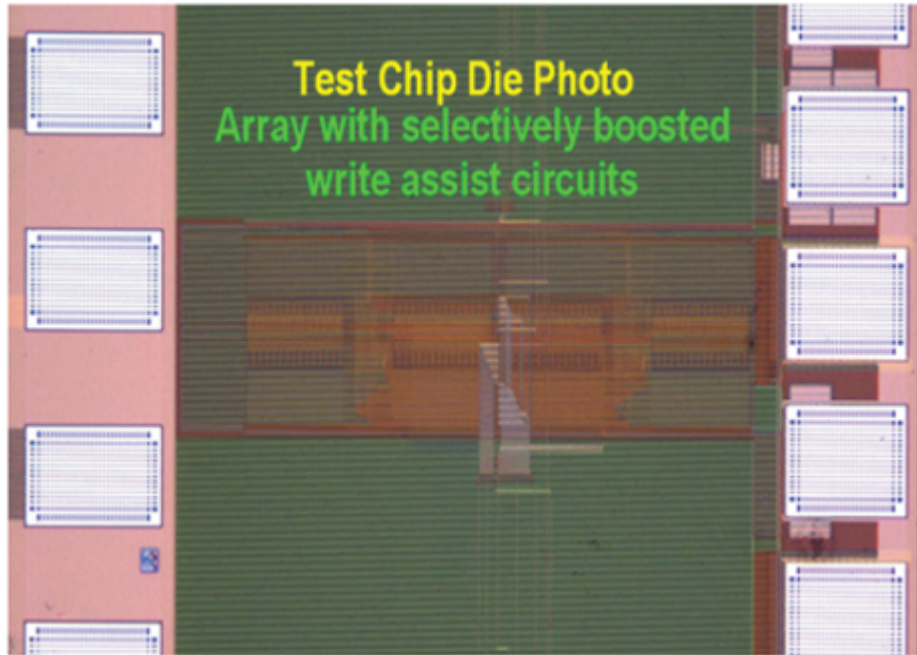


Figure 8.5: Snapshot of the 14nm FinFET SOI technology Die [9].

## 8.3 Ordered Feature Selection Methodology

This section illustrates the accuracy of the proposed logistic regression based method both during the model building phase, and the importance sample points prediction phase in comparison to SPICE circuit simulations. It also presents the final results in terms of the ability of the methodology to estimate the yield compared to pure simulation approaches. At the circuit simulation level we report the results in terms of the false positives and false negatives errors assuming an ideal logistic regression threshold 0.5.

### 8.3.1 Phase1: Model Building

In this section, we focus on the model building phase based on phase 1 data results. Our objective is to demonstrate the cross-validation error trend as the

## Phase 1: Average Cross-Validation Error

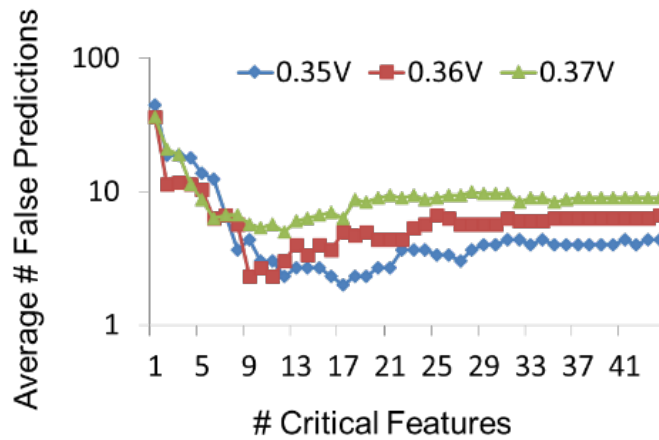


Figure 8.6: Average number of false predictions versus number of Critical Features for the test set [1]

number of features increases and hence the effect of overfitting becomes obvious (which is mostly clear for the experiments labeled 0.36V in Fig. 8.6). Figure 8.6 shows example cross-validation error computed in terms of the average number of false predictions for the negative-boost based design as function of the number of critical features. The number of false prediction is the sum of both false positives and false negatives combined for the test set (size between of 200-250 sample points) in the k-fold cross-validation for the various data sets. Figure 8.7 illustrates the number of critical features corresponding to minimum cross-validation error for both standard and boosted designs.

- Experiments 1 to 6 correspond to the Boosted circuit results for  $V_{dd}=[0.35V-0.4V]$ , and
- Experiments 7 to 10 belong to the standard circuit results for  $V_{dd}=[0.4V-0.43V]$ .

The cross validation error rate for the best features set for the different exper-

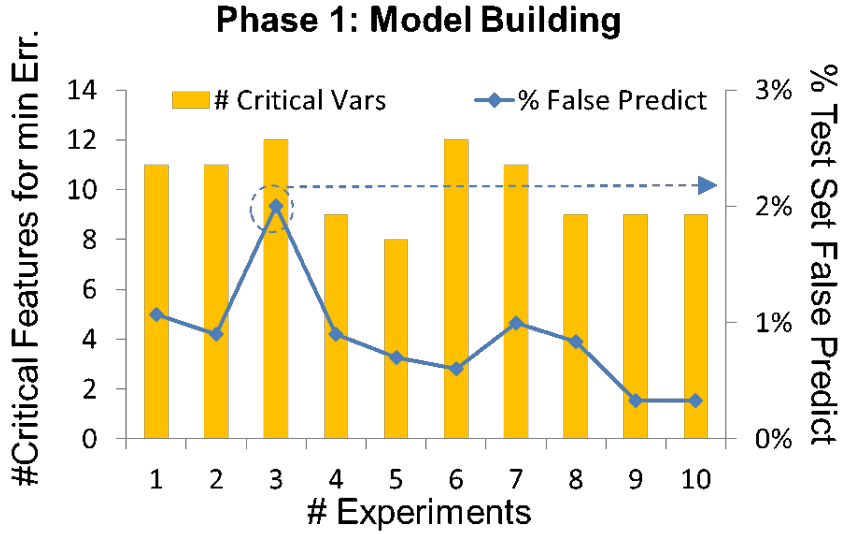


Figure 8.7: Number of critical features for minimum cross-validation error, and corresponding test set error. Experiments 1 to 6 represent negative bitline boost write-assist based circuit results over  $[0.35V-0.4V]$ , and experiments 7 to 10 represent the standard circuit experiments over  $[0.4-0.43V]$  [1].

iments is reported to be less than 2%, and it corresponds to a number of critical features ranging from 8 to 12 features.

### 8.3.2 Phase 2: Importance Samples Prediction

In this section, we report the importance sample phase prediction capability. The model developed in phase 1 using the uniform sampled data is utilized to predict the outcome of the samples generated in phase 2 via the importance sampling distribution. Hence, we use the models obtained in phase 1 to predict phase 2 pass/fail criteria. Each experiment corresponds to a different Vdd value and a number of sample points ranging between 600-1000. Figures 17 and 18 present the results for both designs.

We compare the outcomes to a pure circuit simulation based approach; it can be observed that the number of false predictions is very low with a maximum

**Standard Design : Phase 2 Importance Sample Points Prediction**

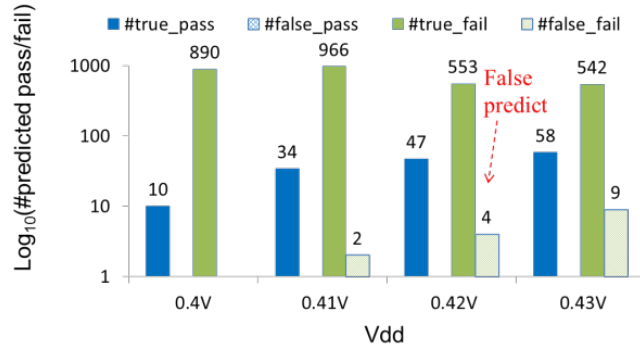


Figure 8.8: Standard Design phase 2 pass/fail prediction for the importance sample false negatives(false pass) were reported for this set of experiments [1]

**Boosted Design: Phase 2 Importance Sample Points Prediction**

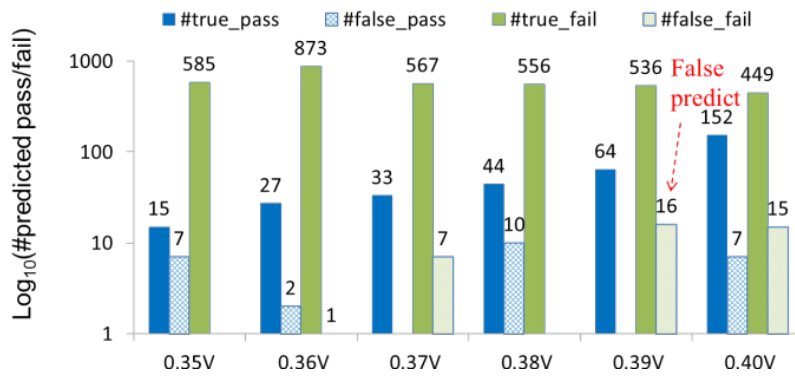


Figure 8.9: Write Assisted Boosted design phase 2 pass/fail prediction for the importance sample points [1].

recorded value of 22 false predictions out of 600 sample points at 0.4V for the boosted circuit, and an average false prediction rate of less than 4.5 for all the experiments. This proves the efficacy of the proposed methodology. It is worth noting that no false negatives were reported for the standard design experiments

### 8.3.3 Yield Estimation

Yield is computed using the importance sample points outcomes obtained in phase estimation. Importance sampling weights are computed using equation 8.3 to approximate the unbiased yield estimates [141].

$$P_f = E(x) = \int I(x) pdf(x) dx \quad (8.3)$$

Figure 8.10 presents fail probability log-plots for the different experiments. It compares the probabilities obtained using full circuit simulation based approach to those obtained using the proposed methodology. The results demonstrate excellent matching all the way down to probabilities of the order 1e-12. This is further emphasized in the yield plots of Figures 20 and 21 where we observe less than 3% difference in the sigma yield numbers; sigma yield numbers are computed using inverse Gaussian cdf function of the fail probabilities. Note that the yield numbers span a wide range of sigma values. Finally, the same Vmin values are reported by our methodology compared to the traditional methodology. We also find around 50mV similar drop in Vmin between the standard and boosted designs as was reported earlier in [9].

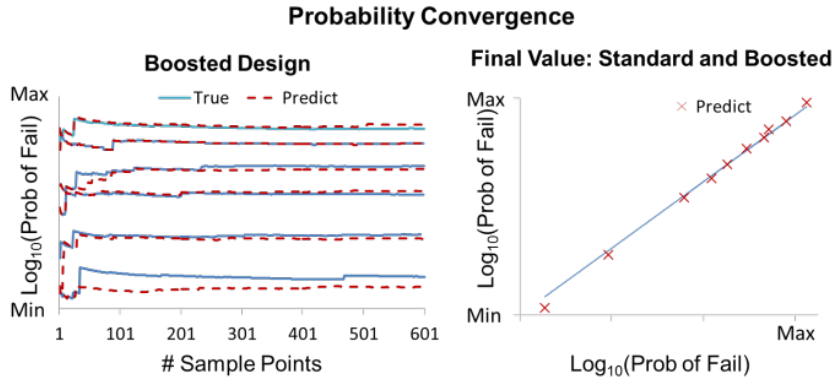


Figure 8.10: normalized for max and min fail probabilities. True: represents circuit simulation based approach. Predict: represents proposed methodology [1]

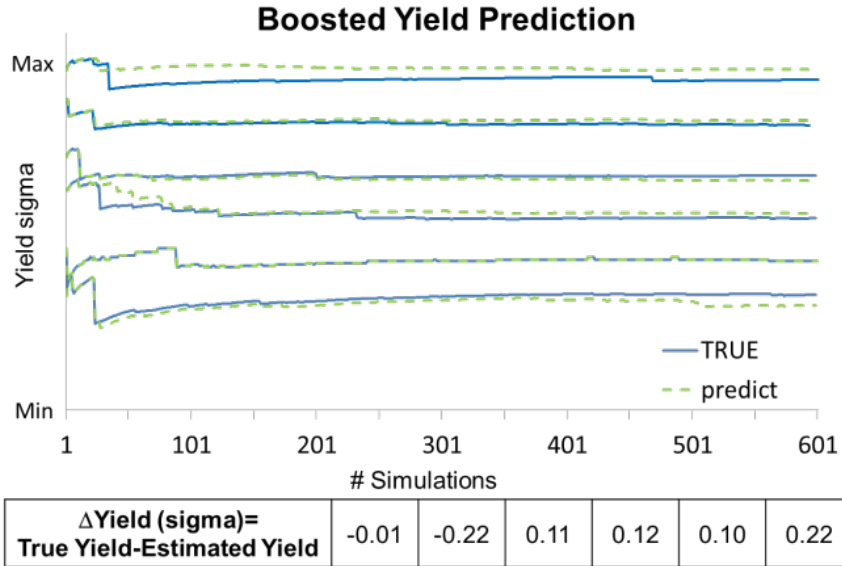


Figure 8.11: Probability Convergence comparison. Results are normalized for max and min fail probabilities. True: represents circuit simulation based approach. Predict: represents proposed methodology [1]

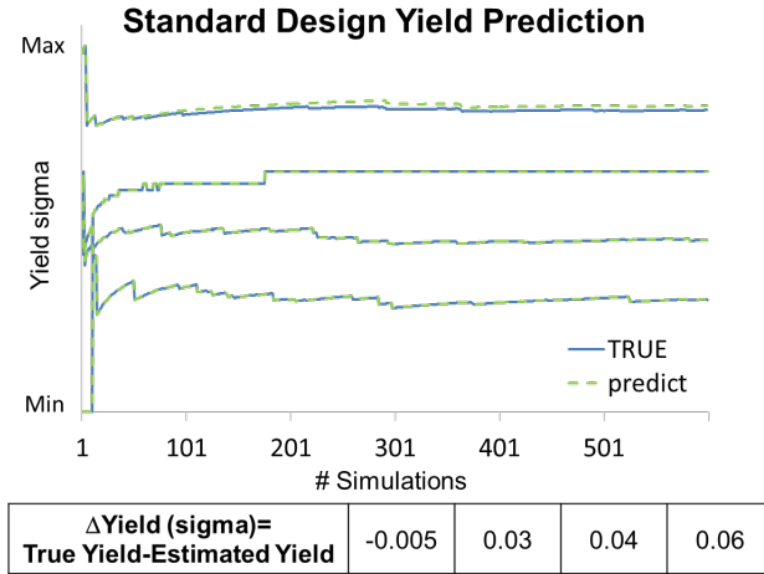


Figure 8.12: Yield Prediction for Standarddesign. Results are normalized for max and min yield. True: represents circuit simulation based approach. Predict: represents proposed methodology [1]

## 8.4 Data Imbalance Handling Approaches

In this section, we review data balancing methods, while taking into consideration the requirements for a conservative yield estimate. First, we introduce some definitions [142].

### 8.4.1 Definitions

Confusion Matrix: A contingency table can be used to evaluate the model performance as the output is binary. This is also known as the Confusion Matrix as illustrated in figure 8.13. For the problem that we are addressing we refer a fail sample point obtained via circuit simulations to an actual positive. The aim is to have a model to predict the importance sampling stage sample points correctly.

The recall rate is another important metric that evaluates the completeness of the classifier. It reflects on the number of the actual positive points that were

**Confusion Matrix**

<b>PREDICTED \ ACTUAL</b>	<b>POSITIVE</b>	<b>NEGATIVE</b>
<b>POSITIVE</b>	<b>(TP)</b> TRUE POSITIVE	<b>(FP)</b> FALSE POSITIVE
<b>NEGATIVE</b>	<b>(FN)</b> FALSE NEGATIVE	<b>(TN)</b> TRUE NEGATIVE

Figure 8.13: Confusion Matrix [6]

correctly labelled by the model 8.4. Hence, it is important to identify a data handling technique that enhances the recall rate in order not to mislabel any of the actual positives. This ensures that a yield estimate reflects the true fail region and prevents being optimistic.

$$Recall = \frac{TP}{TP + FN} \tag{8.4}$$

On the other hand, the precision rate is a metric that evaluates the exactness of the model. It mirrors how many of those predicted positive points were in the original dataset true positives. Accuracy is reflected by a good precision rate. In the context of yield, this means we are not being too pessimistic.

$$Precision = \frac{TP}{TP + FP} \tag{8.5}$$

We employ the data balancing methods [143] that were discussed in Chapter 6 for the purpose of yield analysis. For all our experiments, data balancing generated equally sized minority and majority sample points in the data sets. Figure 8.14 shows an example of an original imbalanced data set and the dataset itself after applying smote. The models built after employing data balancing had a lower number of False Negatives by up to 80% compared to imbalanced approaches.

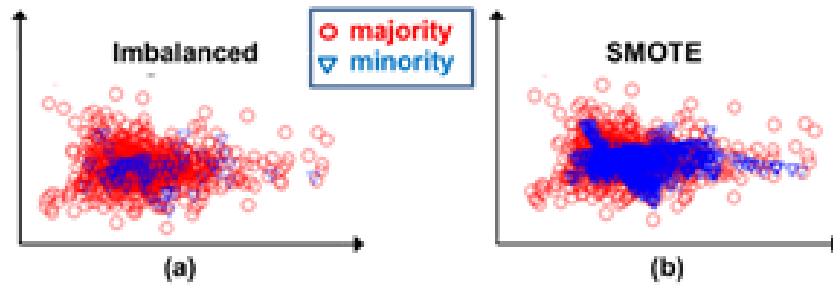


Figure 8.14: Data sample points projected to a 2-D quadratic feature space for (a) the unbalanced original data set and (b) set oversampling via SMOTE [6]

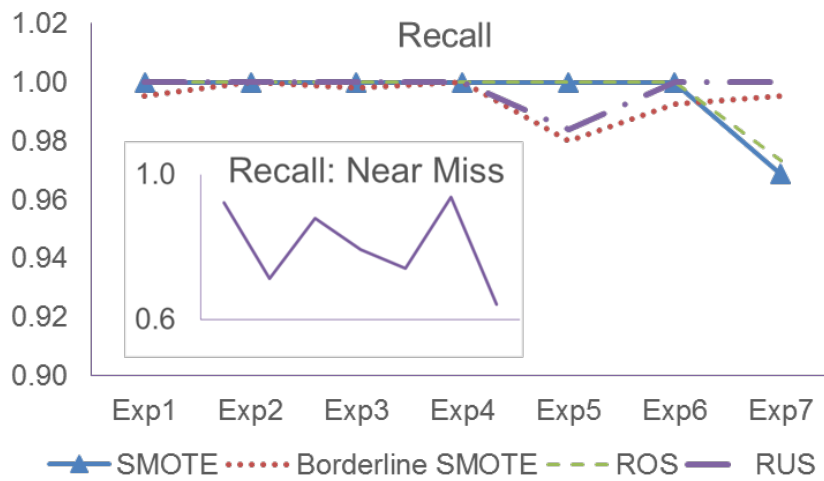


Figure 8.15: Recall rate for the different data balancing techniques [6].

This is shown in Table 8.2. This improved the recall rates. Specifically, for SMOTE, ROS and RUS, most of the experiments had no False Negatives, and hence can be considered conservative. SMOTE had the best recall and precision rates. This was followed by ‘Borderline+SMOTE’, which however was unable to remove all False Negatives because it generates new points near the endangered boundary points rather than emphasizing the entire fail region. The NearMiss method had low recall rates as 0.65 and was not considered for the precision plots. This may have been the result of the border region being very wide when data in the first phase had a small minority ratio.

Table 8.2: The number of False Negatives decreased for most methods (except NearMiss) compared to the imbalanced data set for stage 2 data prediction. NearMiss systematically got worse suffered as stage 1 data imbalance increased.

Method	%FN change to imbalanced data
Borderline+Smote	58↓
NearMiss	>2x increase
Smote	73↓
ROS	77↓
RUS	83↓

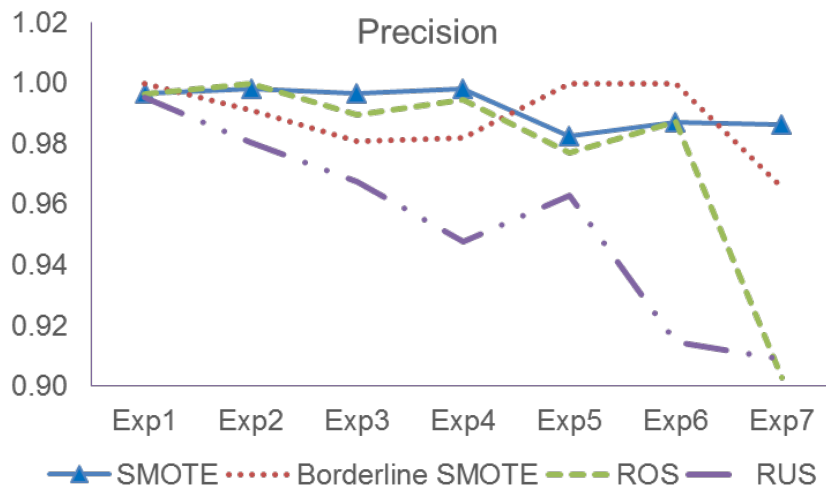


Figure 8.16: Precision rates for the different data balancing techniques.

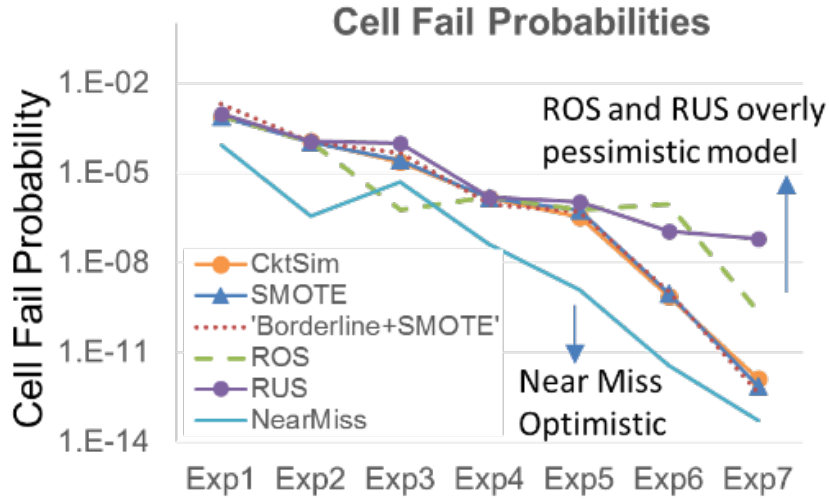


Figure 8.17: Cell probability for thr different data balancing methods [6].

### 8.4.2 Impact of Model Error on Yield Estimation: Imbalanced vs. Balanced Data Sets

Figure 8.17 shows the cell  $P_f$  for the different data balancing methods by using model prediction on stage 2 data set. The results conform with the computed precision and recall rates obtained earlier. ROS and RUS models had good recall rates, but bad precision. This is attributed to the fact taht they had a large number of False Positives and resulted in a pessimistic  $P_f$  estimate. As for the NearMiss methods, the corresponding model had a large number of False Negatives and caused an optimistic estimate for the cell  $P_f$ . It is clear that SMOTE and 'Borderline+SMOTE' are closest to the results obtained from circuit simulation based results.

Furthermore, we compare these results to the imbalanced data set based approach. In order to perform yield analysis obtained using the model developed using imbalanced, Smote, and 'Borderline+SMOTE' data sets versus true circuit simulation. Figure 8.18 shows the absolute yield difference for the 3 meth-

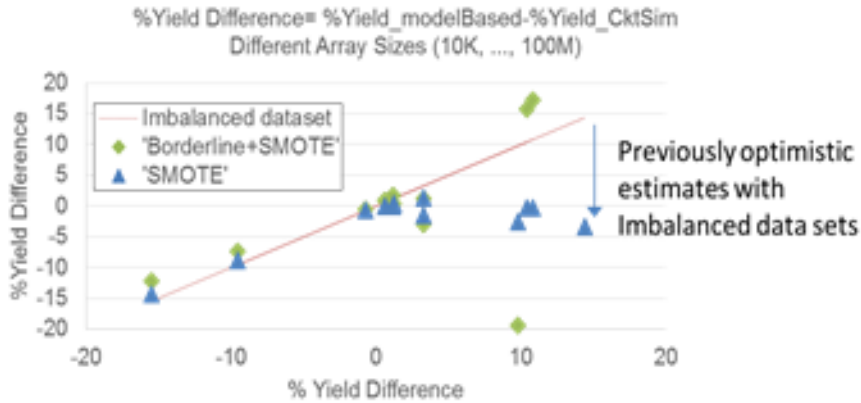


Figure 8.18: Yield difference between the circuit simulation based approach and the yield estimated using logistic regression model based approach for the different balanced/imbalanced data sets. We clearly notice a conservative yield estimate when SMOTE is adopted [6].

ods. ‘SMOTE’ was lowered False Negatives by emphasizing the fail region, and reducing the modeling error that caused an optimistic yield estimates in the imbalanced data set. This ensures accurate and conservative yield estimates. ‘SMOTE+Borderline’, on the other hand, was unable to eliminate all False Negatives consistently. Hence, it provided a fluctuating error. Table 8.3. presents the relative error for the yield estimates.

Table 8.3: Relative Yield Error (for appreciable yield 10%)  $Abs((Yield_{modelBased} - Yield_{CktSim}) / Yield_{CktSim}) * 100$

Imbalanced	SMOTE	SMOTE + borderline
18.20	5.70	20.28
50.29	20.09	80.04

Imbalanced datasets impose several challenges for machine learning algorithms. We proposed to use data balancing methods for fast and accurate statistical analysis methodology of memory designs. Our objective was to achieve accurate and conservative yield estimates. By relying on the SMOTE methodology, we were able to improve the model recall and precision and reduce the

data set induced error on the final yield estimate down to a relative error of 5% compared to 18% for the imbalanced data set-based approaches.

## 8.5 GroupLARS-IRLS

### 8.5.1 Experimental Setup and Yield Analysis

To study the performance of the proposed methodology, we evaluate it in application to a state-of-the-art 14nm FinFET SRAM design with write assist circuitry [144]. As such, the model is trained and validated using a first uniform sampling dataset, and the model is tested in the context of yield analysis on a second importance sampling (IS) data set. The results are evaluated in terms of runtime improvement of IRLS-GroupLARS compared to IRLS-LARS and the classifier is validated in terms of test data accuracy (IS data set) compared to pure circuit-simulations based approach. For purposes of our analysis, we set  $V_{dd} \in [0.35 - 0.45V]$  for the design with and without boosted write assist circuitry for a total of 10 experiments. We apply variability to SRAM cell transistors and the critical path devices resulting in a 45 feature vector after generating interaction and second order polynomial terms [64]. We study the impact of process variations on writeability, which is the ability to flip the cell contents. Hereafter, a positive(negative) sample point represents a simulation that has a writeability fail(pass). We report, for the experiments, the results in terms of recall and precision and yield estimates that span over the range of 2-7  $\sigma$  for the golden simulations.

## 8.5.2 Performance Evaluation

For the IRLS-GroupLARS based yield analysis methodology, we rely on the uniform data sets, which comprised up to  $\sim 2000$  data points each, for the model building phase, and rely on 10-fold cross-validation for finding the best model and critical number of features. For each logistic regression  $\theta$  solution round, we employ 10 nonlinear iterations within which Group-LARS is used to solve the embedded least squares problem for the step direction solution. Figure 8.19 presents the runtime encompassed for the different nonlinear iterations as we search for the critical number of features for an example data set. The runtime improvements for Group LARS increase as the desired number of Features increases. Hence, we record upto 14.7x reduction in runtime when building the model corresponding to the maximum number of features, and upto 6.57x reduction in runtime for the different experiments when 10-fold CV is employed to determine the critical number of features. Note that the advantage is expected scale for experiments with larger dimensions. It is evident that IRLS-GroupLARS approach runtime is almost fixed and does not increase with the number of features due to the fact that each Group LARS LSQ solution invokes two iterations: the first relying on the *critGroup* factor from the previous round and the second searching for the next critical variable based on a weighted least angle direction. Figure 8.20 presents the best model cross-validation error on the uniform data set along with the critical number of features employed by the classifier for the different experiments. We record a maximum  $CVE_{min}$  of 4.57% and average value of 1.13%. We then test the accuracy of the classifier on the importance sample points data set. The metrics rely on the test data false positive (FP), false negative (FN), true positive (TP) and true negative (TN) statistics that are presented in the same Figure. A maximum false prediction rate of  $\sim 3.38\%$  is recorded as indicated in Figure 8.21

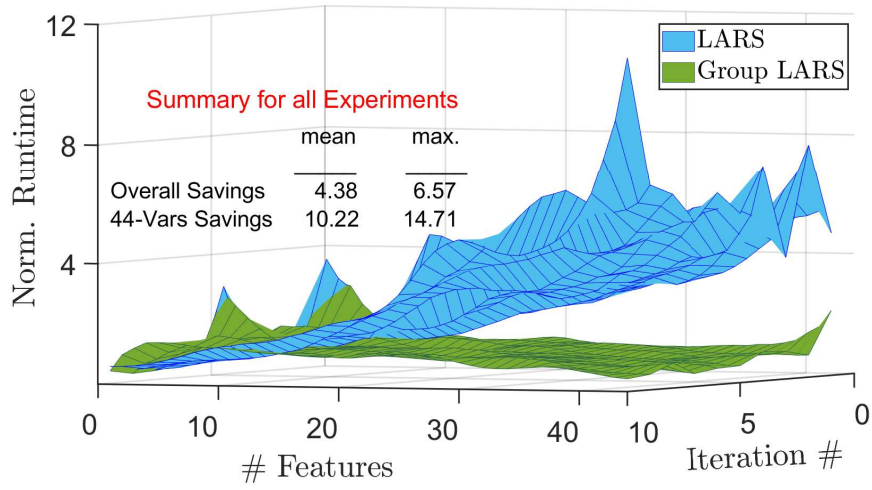


Figure 8.19: Runtime savings for IRLS-GroupLARS compared to IRLS-LARS for an example Experiment. Embedded table summarizes results for the different experiments [5].

which also reports an average accuracy of 98.7%, and very high precision and recall rates. The latter represents the ratio of the true positives to all positive sample points and indicates excellent classifier generalization capability. Finally, the test data outcomes on the importance sample data points are used to predict the unbiased yield estimate of the SRAM design [15] using IRLS-GroupLARS based classifier. Figure 8.22(a) presents the convergence for the fail probability log-plots for the different experiments compared to the pure circuit simulation based approach based on the test data. The results demonstrate excellent matching for the different probability estimates for values as low as  $10^{-12}$ . We then estimate the yield  $\sigma$  values using inverse Gaussian cdf of the fail probabilities. Figure 8.22(b) presents the corresponding  $\sigma$  estimate values where we report a maximum of 0.17  $\sigma$  error for the IRLS-GroupLARS based approach compared to the golden reference, and an insignificant average error of 0.09  $\sigma$ .

We propose a Group LARS based iterative re-weighted least squares methodology for efficient and accurate regularized logistic regression with application

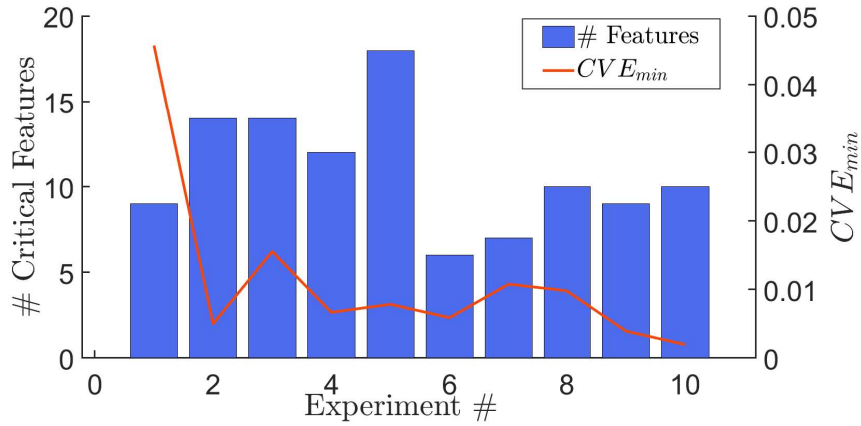
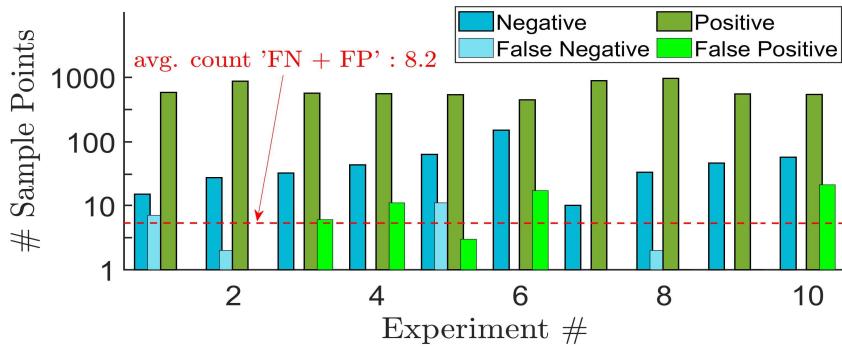


Figure 8.20:  $CVE_{min}$  and the critical number of features [5].



Exp #	1	2	3	4	5	6	7	8	9	10
<b>Precision</b>	1.000	1.000	0.990	0.981	0.994	0.964	1.000	0.999	0.998	0.963
<b>Recall</b>	0.988	0.998	1.000	1.000	0.980	1.000	1.000	0.998	1.000	1.000
<b>Accuracy</b>	0.988	0.998	0.990	0.982	0.977	0.972	1.000	0.997	0.998	0.966

Figure 8.21: Prediction accuracy for IRLS-GroupLARS.

to memory yield analysis. The approach benefits from Group LARS inherent weighted least angle direction to pass and update the critical feature group from one solution round to another thereby speeding up the IRLS solution. For the experiments underway, we demonstrate up to 14x speed up for the model development using the maximum number of desired Features. This is expected to increase even more for very high-dimensional problems. We demonstrate high accuracy for the resulting classifier for the importance sample points both in terms of the number of false predicts as well as the yield estimate.

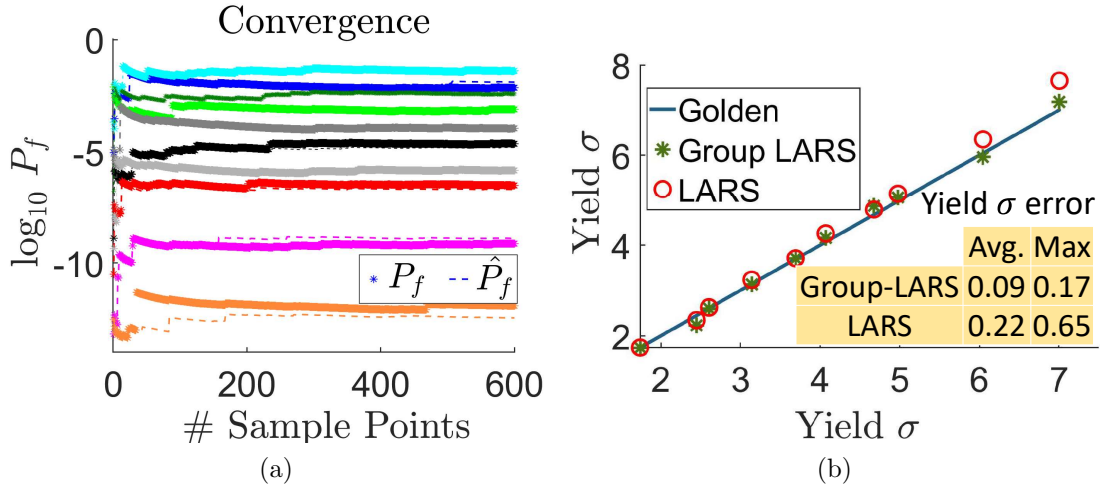


Figure 8.22: (a) Probability Convergence results for  $IRLS_{GroupLARS}$  estimate  $\hat{P}_f$ . Golden ( $P_f$ ) represents circuit simulation based approach. b) Yield estimate [5]

## 8.6 Best Balance Ratio Ordered Feature Selection

### 8.6.1 Performance Evaluation: Robustness and Compactness

We compare the performance of the proposed BBR-OFS methodology to traditional L1-regularized SVM solutions, ordered feature selection using the unbalanced datasets, as well as implicitly balanced two dimensional solution surface weighted SVM methodology. Our golden reference is the pure circuit simulation based approach. For all our experiments, different data balancing ratios were employed, and the BBR-OFS model is used to estimate the yield. The first set of results aims at evaluating the performance of the model. The second set of results aims at evaluating the overall performance of the proposed methodology in the context of yield analysis. Finally, we report on the run-time savings for the proposed methodology. Hereon, we will use the following terminology.

1.  $L1\text{-SVM}_\rho$ : refers to the best solution along the LASSO regularized SVM solution path [21]. The subscript refers to employing the natural distribution ratio,  $\rho$ , of the minority to majority sample points for a given dataset. The original set of features is employed, and sparsity is solely enforced via LASSO regularization.
2.  $\text{OFS}_\rho$ : This approach employs ordered feature selection that iteratively calls  $L1\text{-SVM}$  solutions to identify the best feature set on the natural datasets; i.e., no data balancing is employed.
3.  $\text{BBR-OFS}$ : refers to the proposed best balance ratio ordered feature selection methodology. OFS is employed with different data balancing ratios,  $\rho \leq \beta \leq 1$  for the minority to majority sample points. Data balancing is performed using SMOTE.
4.  $2\text{DWSVM}$ : The algorithmic approach proposed in [145] is employed to manage data balancing implicitly through a 2D  $(\lambda, \pi)$  solution surface, where the  $\pi$  value is used to help emphasize the effect of the minority sample points in the cost function.

Due to the nature of the importance sampling methodology, each experiment is associated with preprocessed data sets corresponding to the two phases of importance sampling: a uniform sampling data set,  $P_{uni}$  and an importance sampling data  $P_{MIS}$  set. The model building phase employs the uniform dataset, and the yield estimation Phase 2 employs the importance sampling dataset. As such, we rely on the uniform data set for model building. We employ 5-fold cross validation and rely on cross-validation error as holistic metric to provide an unbiased evaluation of different classifier models and balancing ratios during this phase.

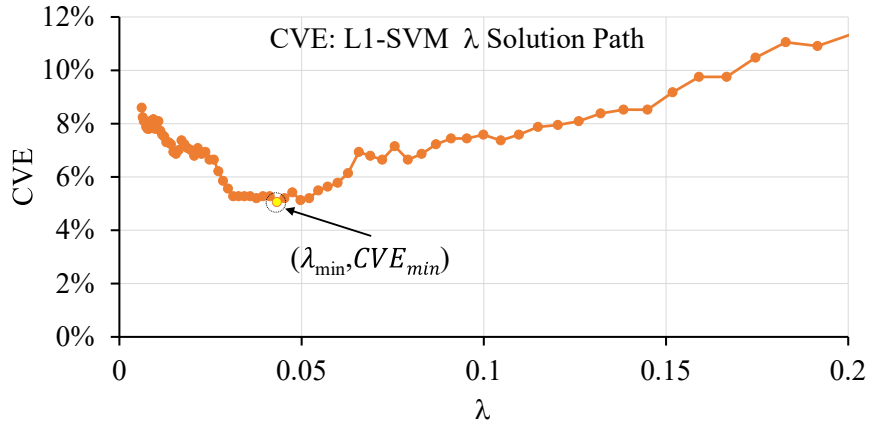


Figure 8.23: Cross Validation error versus  $\lambda$  for an example  $L_1$ -SVM path [7].

We evaluate the performance of the model in terms of the yield estimation using the importance sampling dataset which counts as the test data for our purposes.

Figure 8.23 presents the CVE error as function of  $\lambda$  for an example  $L_1$ -SVM solution path demonstrating the pair  $(\lambda_{min}, CV_{min})$  for one of the experiment. For purposes of our analysis, BBR-OFS relies on the  $L_1$ -SVM path in the search for best ordered feature sets and balance ratios  $\beta_j$ .

Figure 8.24 presents example  $CV E_{min}$  values for the fully balanced and natural uniform sampling datasets for the different experiments when their respective ordered features sets are employed near the optimal  $\Omega_{BBR-OFS}$  solutions.

The impact of SMOTE on the regularization parameter and hence  $CV E_{min}$  is evident with lowest cve dropping close to 2. Our results, as will be demonstrated later, show that ordered feature selection along balancing improve the results significantly thereby addressing model robustness and compactness considerations and improving the prediction accuracy rate both at the model building and yield estimation phases. Figure 8.25 compares the reduction in the number of features (independent variables) for the proposed BBR-OFS methodology versus  $L_1$ -SVM $_{\rho}$ , applied to the natural distribution, and  $L_1$ -SVM $_{\beta_j}$ , is applied without

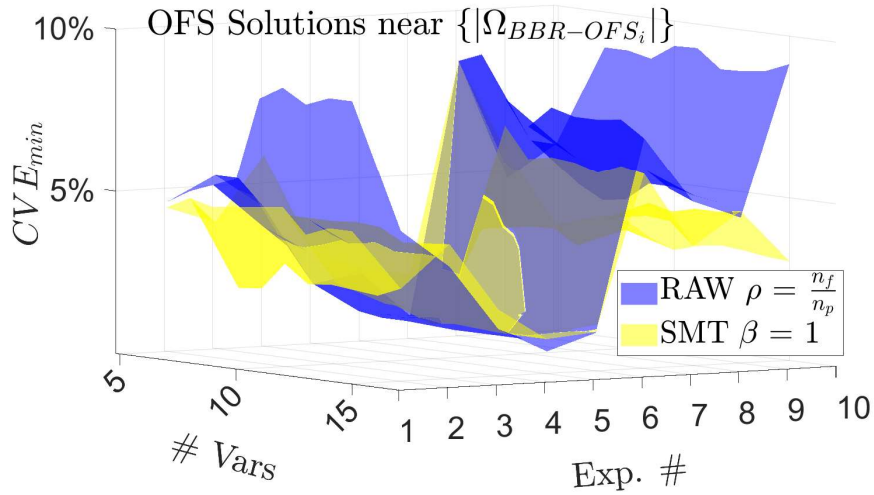


Figure 8.24:  $CVE_{min}$  for the different experiments for the fully balanced and raw ratios for the uniform sampling datasets [7].

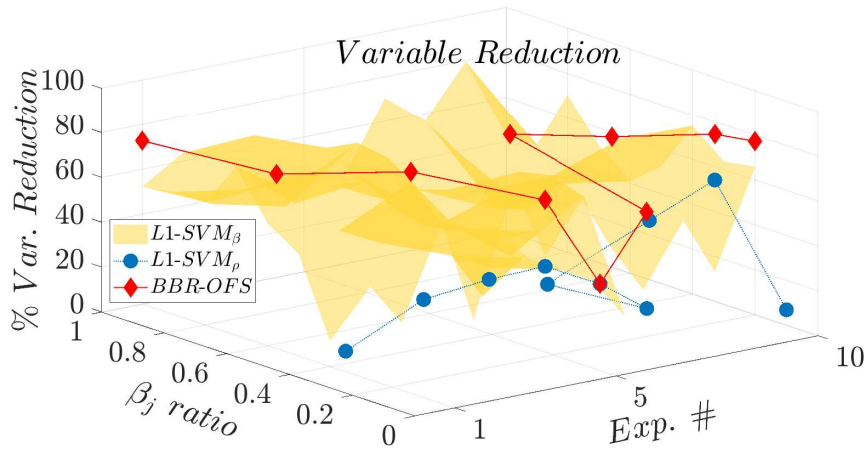


Figure 8.25: Variable reduction for the proposed BBR-OFS models versus is higher than that of  $L1-SVM_{\rho}$ , i.e.,  $L1-SVM$  on raw data and  $L1-SVM_{\beta}$ , i.e.,  $L1-SVM$  on the different balanced datasets.

ordered feature selection to the different balancing ratios,  $\beta_j^i$  where  $\rho^i \leq \beta_j^i \leq 1$ , and  $i \in [1, 10]$  represents the experiment number. We note 32.5% average variable reduction for the  $L1-SVM_\rho$ , compared to 46.7% for the  $L1-SVM_{\beta_j}$  solutions. The proposed BBR-OFS method results in 71.5% reduction in the number of variable. This clearly shows that SMOTE together with ordered feature selection help remove insignificant features thereby enabling a sparse solution and minimizing overfitting.

### 8.6.2 Performance Evaluation: BBR-OFS AUC and ROC

To demonstrate the efficacy of our BBR-OFS solutions, in what follows, we evaluate their respective Receiver Operating Characteristic Curve (ROC) curves. In addition to CVE, ROC analysis has also been used as a metric for imbalanced data sets [146], where the area under the ROC curve (AUC) is used to assess the classifier performance. In fact, it is well established that AUC does not favor one class over the other, and hence it is not biased to the majority class and can be used to compare the performance of the different classification methods for imbalanced datasets. For any classifier, there are four possible prediction outcomes:

1. True positive (TP): correctly predicted minority point.
2. False positive (FP): incorrectly predicted minority point.
3. True negative (TN): correctly predicted majority.
4. False negative (FN): incorrectly predicted majority

The ROC curve depicts the trade-off between the true positive rate (TPR) and the false positive rate (FPR) to evaluate the performance of the binary classifier

[147]. TPR is referred to as sensitivity and represents the fraction of correctly predicted minority sample points from all the minority sample points as defined in Eqn. (8.6). FPR is referred to as specificity and reflects the fraction of minority sample points that are mispredicted as presented in Eqn. (8.7).

$$TPR = \frac{TP}{TP + FN} \quad (8.6)$$

$$FPR = \frac{FP}{FP + TN} \quad (8.7)$$

It has been shown, that for a given classifier, AUC can be viewed as the probability of correctly ranking the class “positive”-“negative” sample point pairs. AUC averages the score of a minority (positive) sample point having a higher probability than a majority (negative) sample point for all between-class pairs. Hence, AUC can be computed as the proportion of all random pairs which are ranked correctly by the classifier. It is thus viewed as a rank statistic that can be computed according to Mann-Whitney Wilcoxon test in Eqn. (8.8) without the need for building the ROC curve.

$$AUC = \frac{1}{n_f n_p} \sum_{i=1}^{n_p} \sum_{j=1}^{n_f} I(f(x_i^+) > f(x_j^-)) \quad (8.8)$$

where  $I(f(x_i^+) > f(x_j^-))$  is the indicator function that evaluates to one when the embedded inequality holds. For our purposes, TPR corresponds to detecting fail sample points accurately, and a high TPR is critical for the yield estimate not being too optimistic. A high FPR is not desired as it results in a pessimistic yield estimate, especially that FP’s can be close to the fail boundary. Thus, the higher the AUC the better are the results, and the ideal AUC value is 100%. A 50% AUC value indicates that the classifier has bad predictability that is equivalent

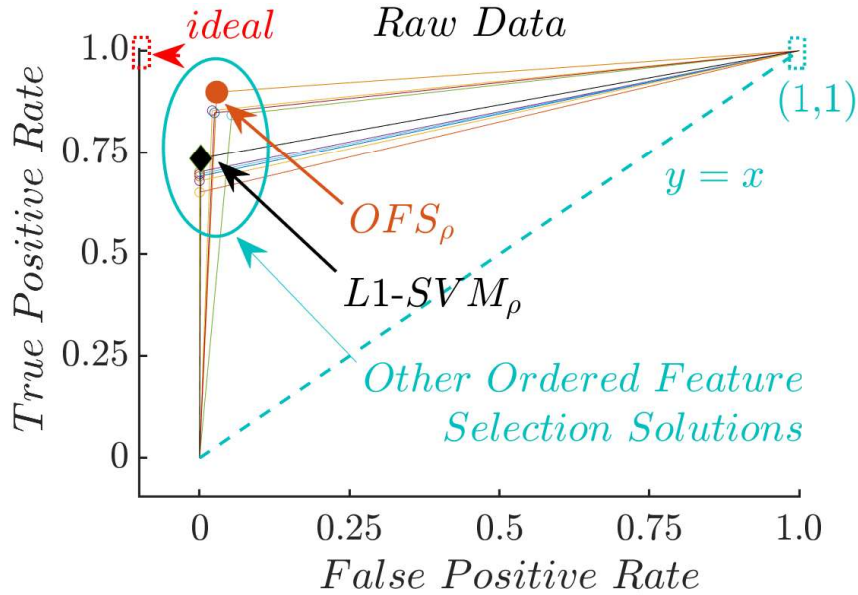


Figure 8.26: ROC curves for an example raw data set indicate improved AUC value for the  $OFS_\rho$  solution compared to  $L1-SVM_\rho$ . AUC values for the ordered feature selection iterations are also presented [7].

to that of tossing a coin.

Figure 8.26 presents the ROC curves for OFS applied to the raw data for one experiment. It is clear that the AUC for the  $OFS_\rho$  solution is better than that for the  $L1-SVM_\rho$  solution. Its TPR ratio is close to 90% compared to 75% for  $L1-SVM_\rho$ , with 89% and 73% AUC values for the  $OFS_\rho$  and  $L1-SVM_\rho$  respectively. The figure also depicts the ROCs for several solutions obtained from the iterations of OFS. It is clear, that with the same underlying dataset, OFS achieved better performance than the standard  $L1-SVM$  method. Figure 8.27 presents the AUC values for the BBR-OFS solution when OFS is applied to the different balancing ratios. The figure thus presents the AUC values for OFS iterations for two groups which overlap in some cases: 1) Best solution group: OFS solutions where each data set is balanced by its respective best balance ratio as obtained from the BBR-OFS solution, and 2) the raw data set OFS solutions including  $L1-SVM_\rho$ .

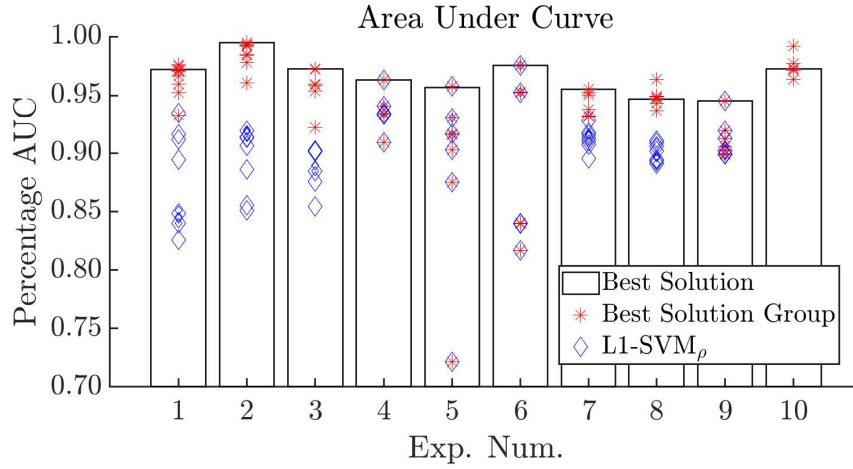


Figure 8.27: The AUC value for the BBR-OFS solution is among the best compared to the raw data set solutions and the other solutions employing the same data balancing ratio with varying feature sets [7].

Note for our BBR-OFS solutions, only one experiment favored a fully balanced solution, five favored an intermediate balancing ratio and four sets favored the raw distribution OFS solution. For the raw datasets, we observe AUC values fluctuate in some cases lower than 70% for the  $L1-SVM_{\rho}$  solution. Clearly, OFS along with optimal balancing ratio help shrink this uncertainty and attain better predictive accuracy based. BBR-OFS solution often ranked highest among its group with the balanced datasets demonstrating tighter AUC ranges. As discussed earlier, we view AUC as the probability of the ability of a model to correctly rank the classes. Thus, the figure clearly illustrates the advantage of BBR-OFS in enhancing the AUC values and hence the generalization capability of the classifier.

### 8.6.3 Implicit versus Explicit data Balance Ratios: '2D WSVM'

For the SMOTE data balancing, we varied  $\beta_j \in [\rho, 1]$ . The 2DWSVM approach explores the  $[\lambda, \pi]$  two dimensional solution surface to address the data imbalance

and enable efficient exploration of the piecewise linear parameter solutions in the  $[\lambda, \pi]$  space. We ran the same datasets through the '2DWSVM' approach and identified the best solution pair  $[\lambda, \pi]$ . Figure 8.28 presents the raw ratio  $\rho$  compared to the best solution  $\pi$ -adjusted ratio as presented in Eqn. (8.9) where the majority are weighted by  $\pi$  and the minority by  $(1 - \pi)$ . We notice that the best solution adjusted ratio was always close to one, with best solution  $\pi$  in solution surface space taking a value close to the raw distribution. The best solution CVE ranged between 6% and 25% with an average of 18% for the different experiments, and the CVE error was highest for the datasets with very low  $\frac{n_f}{n_p}$  ratios.

$$\left(\frac{n_f}{n_p}\right)_{\pi_{adj}} = \frac{(1 - \pi) * n_f}{\pi * n_p} \quad (8.9)$$

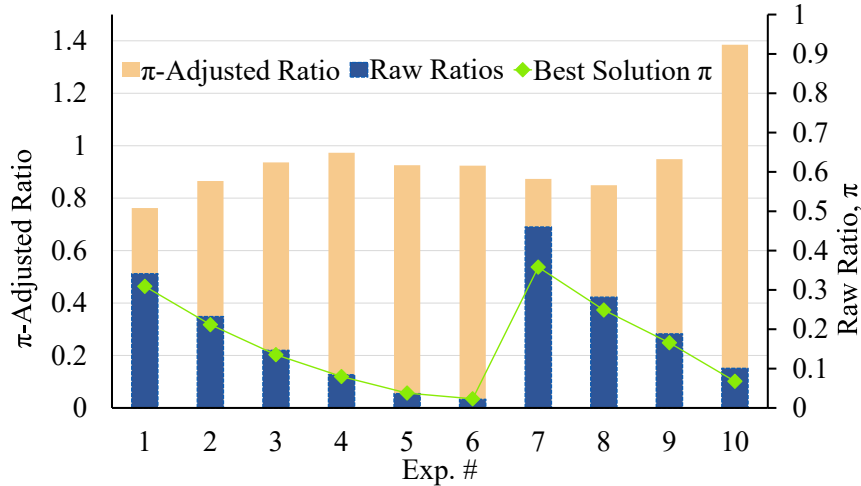


Figure 8.28: 2DWSVM  $\pi$ -Adjusted ratio for the best solution in 2D solution surface space [7].

#### 8.6.4 Yield Estimation and Runtime

We estimate the yield using the importance sample points. For each dataset, we computed the rare event failure probability  $P_f$  for the specific failure criteria using the Phase 2 performance metric values. We represented  $P_f$  by its equivalent  $\sigma$  value on the standard normal distribution:

$$\sigma = \phi^{-1}(1 - P_f) \quad (8.10)$$

where  $\phi$  is the standard normal cumulative density function. Herein, we compare the accuracy and the efficiency of the proposed BBR-OFS methodology for rare fail event estimation. Our reference is the pure circuit simulation based approach. The BBR-OFS based yield estimate outperformed the other model-based estimations as emphasized in Figures 8.29 and 8.30 below. Figure 8.29 presents the absolute value of yield error  $|\sigma_{err}|$  which is the difference of the yield sigma value between that obtained from the pure circuit simulation based approach,  $\sigma_{ref}$ , and that obtained by relying on the different models for predicting the circuit performance,  $\sigma_{hat}$ . The implicit *2DWSVM* demonstrated the highest error as was expected based on what was reported earlier in terms of CVE values.  $OFS_\rho$  outperformed  $L1-SVM_\rho$  for the yield estimate inline to what has been demonstrated in terms of the AUC-ROC analysis for the model prediction capability. Finally, BBR-OFS also demonstrated both improved average and worst case error yield sigma errors compared to  $OFS_\rho$ . Figure 8.30 presents the BBR-OFS solution based yield estimate compared to the golden reference citing on average an error in the yield sigma estimate of around 0.19 with maximum error below  $0.5 \sigma$ . This is compared to an average sigma error of 0.36 and 0.49 for the  $OFS_\rho$  and  $L1-SVM_\rho$  with a maximum error of  $1.5 \sigma$  for both. This clearly highlights

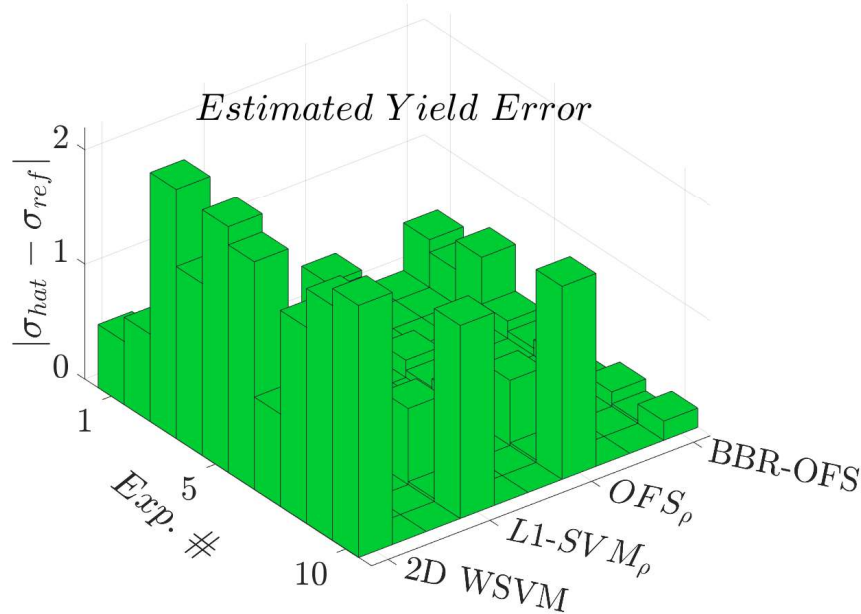


Figure 8.29: Yield error,  $|\sigma_{err}| = |\sigma_{hat} - \sigma_{ref}|$ , where  $\sigma_{ref}$  is the yield for the pure-circuit simulation based approach.  $\sigma_{hat}$  is the yield corresponding to the model based importance sample points evaluation for the different approaches [7].

the advantages of data balancing together with ordered feature selection. From the perspective of runtime complexity, the explicit methods offered clear speedup advantages as compared to the pure circuit simulation based approach as they help reduce the required runtime by completely eliminating the need for circuit simulations in the importance sampling phase. For our experiments the model building was performed on the uniform dataset which comprised upto  $\sim 2000$  data points for the balanced datasets depending on the raw balance ratio, and the yield estimation was performed on close to 1000 importance sample points on a large memory cross-section. Model building in R and SPICE simulations were performed on a 4GHz IBM Power7 core processor machine. The BBR-OFS methodology demonstrated  $\sim 179x$  speedup for phase 2 simulations thereby enabling significant speedup which is much needed when simulating large cross-

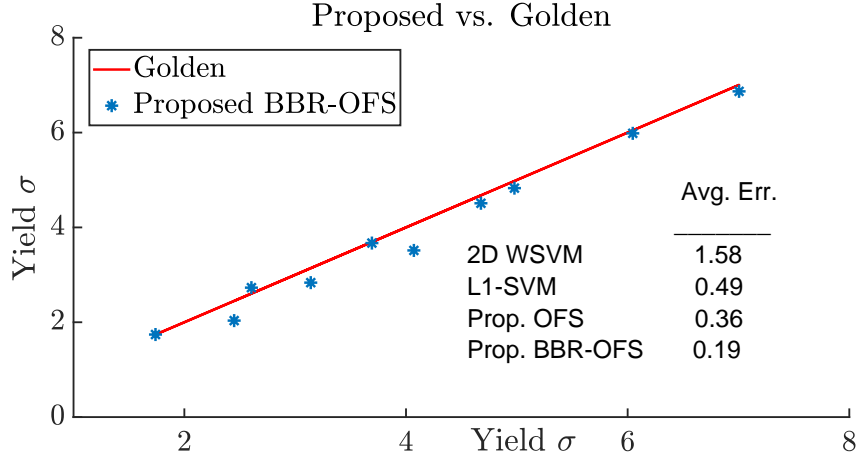


Figure 8.30: Scatter yield plot for pure circuit simulation based approach (golden), versus BBR-OFS (proposed). BBR-OFS demonstrates an average yield sigma error of 0.19 sigma [7].

sections. Speedup for  $L1$ -SVM and  $OFS_\rho$  is also presented and is demonstrated to be higher due to the fact that BBR-OFS involves searching for the best balance ratios and that SMOTE enlarges the training datasets. For  $L1$ -SVM we report 0.65 seconds of runtime for the model building (fitting) using R, and similar values for the importance sample points model evaluation. The implicit 2DWSVM based approach lagged from a runtime perspective due to the cost of the model building phase which seemed to suffer in terms of the number of sample points and due to the 2D space explorations. In conclusion, we observe that our proposed BBR-OFS approach presents clear benefits both in terms of yield accuracy and runtime efficiency compared to the pure-circuit simulation based approach as well as other approaches.

We proposed a best balance ratio ordered feature selection methodology that aims at boosting classifier performance in the context of data imbalance aware rare fail event estimation. We tackle the ordered feature selection problem as a two-level optimization framework approximation to the  $L_0$ -norm regularization

Table 8.4: Phase II Speedup and Runtime in seconds.

Phase	Ckt Sim	2D WSVM	L1-SVM	OFS	BBR-OFS
Uniform Sampling	18000	18000	18000	18000	18000
Model Fitting	N/A	Several hrs	0.65	19.39	150.43
Importance Sampling	27000	0.22	0.45	0.4	0.32
Phase II Speedup	1	0.5x	24421x	1364x	179x

and we rely on synthetic minority oversampling technique as a popular data balancing method in the ML paradigm for probing for the best balance ratio. We investigate the accuracy and performance trade-offs of the proposed methodology when estimating the yield of an industrial 14nm FinFET SRAM design. Our results demonstrate that BBR-OFS outperforms other methods and offers an efficient estimate of the memory reliability metric with an average error of 0.19 sigma for the yield estimate and runtime improvement of 179x compared to the pure-circuit simulation based approach.

# Chapter 9

## Discussion and Future Work

This thesis work introduces novel machine learning methodologies spanning regression and classification. It introduces rigorous merits that are bound to signify the strength of the proposed models. It (1) presents a regularized logistic regression based methodology (2) proposes data balancing methods for imbalanced datasets (3) introduces a Group-LARS based iterative reweighted least squares method and (4) compares various data balancing ratios and (5) introduces best balance ratio technique along with ordered feature selection to improve model generalization capability. The thesis proves the efficiency of the proposed methods by achieving high accuracy for the models and excellent yield estimation. The proposed methods differ from existing methods in that they are based on importance sampling as an estimation phase and that they target algorithmic improvements for enhanced runtime, accuracy and efficiency for imbalance datasets encountered for memory design yield modeling problems.

# Bibliography

- [1] L. Shaer, R. Kanj, R. Joshi, M. Malik, and A. Chehab, “Regularized logistic regression for fast importance sampling based sram yield analysis,” in *2017 18th International Symposium on Quality Electronic Design (ISQED)*, pp. 119–124, 2017.
- [2] R. Kanj, R. V. Joshi, L. Shaer, A. Chehab, and M. Malik, “Fast statistical analysis using machine learning,” in *Machine Learning in VLSI Computer-Aided Design*, pp. 323–348, Springer, 2019.
- [3] D. F. Findley, “Model selection: Akaike’s information criterion,” *Wiley StatsRef: Statistics Reference Online*, 2014.
- [4] R. Kohavi *et al.*, “A study of cross-validation and bootstrap for accuracy estimation and model selection,” in *Ijcai*, vol. 14, pp. 1137–1145, Montreal, Canada, 1995.
- [5] L. Shaer, R. Kanj, R. Joshi, and A. Chehab, “Group lars based iterative reweighted leastsquares methodology for efficient yield modeling,” *Submitted for Revision*.
- [6] L. Shaer, R. Kanj, and R. Joshi, “Data imbalance handling approaches for accurate statistical modeling and yield analysis of memory designs,”

- in *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, IEEE, 2019.
- [7] L. Shaer, R. Kanj, and R. Joshi, “1a best balance ratio ordered feature selection methodology for robust and fast statistical analysis of memory designs,” *Submitted for Revision*.
- [8] M. Malik, R. V. Joshi, R. Kanj, S. Sun, H. Homayoun, and T. Li, “Sparse regression driven mixture importance sampling for memory design,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 1, pp. 63–72, 2017.
- [9] R. V. Joshi, M. Ziegler, H. Wetter, C. Wandel, and H. Ainspan, “14nm finfet based supply voltage boosting techniques for extreme low v min operation,” in *2015 Symposium on VLSI Circuits (VLSI Circuits)*, pp. C268–C269, IEEE, 2015.
- [10] S. Nassif, “Delay variability: sources, impacts and trends,” in *2000 IEEE International Solid-State Circuits Conference. Digest of Technical Papers (Cat. No. 00CH37056)*, pp. 368–369, IEEE, 2000.
- [11] X. Li and H. Liu, “Statistical regression for efficient high-dimensional modeling of analog and mixed-signal performance variations,” in *Proceedings of the 45th annual Design Automation Conference*, pp. 38–43, 2008.
- [12] A. Singhee and R. A. Rutenbar, “Statistical blockade: Very fast statistical simulation and modeling of rare circuit events and its application to memory design,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 8, pp. 1176–1189, 2009.

- [13] K. Agarwal and S. Nassif, “Statistical analysis of sram cell stability,” in *2006 43rd ACM/IEEE Design Automation Conference*, pp. 57–62, 2006.
- [14] S. Mukhopadhyay, H. Mahmoodi, and K. Roy, “Statistical design and optimization of sram cell for yield enhancement,” in *IEEE/ACM International Conference on Computer Aided Design, 2004. ICCAD-2004.*, pp. 10–13, 2004.
- [15] R. Kanj, R. Joshi, and S. Nassif, “Mixture importance sampling and its application to the analysis of sram designs in the presence of rare failure events,” in *2006 43rd ACM/IEEE Design Automation Conference*, pp. 69–72, 2006.
- [16] J. Yao, Z. Ye, and Y. Wang, “Efficient importance sampling for high-sigma yield analysis with adaptive online surrogate modeling,” in *2013 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 1291–1296, 2013.
- [17] S. Sun, Y. Feng, C. Dong, and X. Li, “Efficient sram failure rate prediction via gibbs sampling,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 12, pp. 1831–1844, 2012.
- [18] L. Dolecek, M. Qazi, D. Shah, and A. Chandrakasan, “Breaking the simulation barrier: Sram evaluation through norm minimization,” in *2008 IEEE/ACM International Conference on Computer-Aided Design*, pp. 322–329, 2008.
- [19] I. M. Elfadel, D. S. Boning, and X. Li, *Machine learning in VLSI computer-aided design*. Springer, 2019.

- [20] H. Chen and D. Boning, “Online and incremental machine learning approaches for ic yield improvement,” in *2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 786–793, 2017.
- [21] C. Yi and J. Huang, “Semismooth newton coordinate descent algorithm for elastic-net penalized huber loss regression and quantile regression,” *Journal of Computational and Graphical Statistics*, vol. 26, no. 3, pp. 547–557, 2017.
- [22] S. J. Shin, Y. Wu, and H. H. Zhang, “Two-dimensional solution surface for weighted support vector machines,” *Journal of Computational and Graphical Statistics*, vol. 23, no. 2, pp. 383–402, 2014.
- [23] M. Mani, A. Devgan, and M. Orshansky, “An efficient algorithm for statistical minimization of total power under timing yield constraints,” in *Proceedings of the 42nd annual Design Automation Conference*, pp. 309–314, 2005.
- [24] F. Girosi, M. Jones, and T. Poggio, “Regularization theory and neural networks architectures,” *Neural computation*, vol. 7, no. 2, pp. 219–269, 1995.
- [25] K. Singhal and J. Pinel, “Statistical design centering and tolerancing using parametric sampling,” *IEEE Transactions on Circuits and Systems*, vol. 28, no. 7, pp. 692–702, 1981.
- [26] J. Ramaley, “Buffon’s noodle problem,” *The American Mathematical Monthly*, vol. 76, no. 8, pp. 916–918, 1969.
- [27] N. Metropolis and S. Ulam, “The monte carlo method,” *Journal of the American Statistical Association*, vol. 44, no. 247, pp. 335–341, 1949.

- [28] D. G. Horvitz and D. J. Thompson, “A generalization of sampling without replacement from a finite universe,” *Journal of the American statistical Association*, vol. 47, no. 260, pp. 663–685, 1952.
- [29] E. Veach, *Robust Monte Carlo methods for light transport simulation*, vol. 1610. Stanford University PhD thesis, 1997.
- [30] J. E. Gentle, *Random number generation and Monte Carlo methods*. Springer Science & Business Media, 2006.
- [31] P. Dutre, P. Bekaert, and K. Bala, *Advanced global illumination*. CRC Press, 2018.
- [32] E. George, W. G. Hunter, and J. S. Hunter, *Statistics for experimenters: Design, innovation, and discovery*. Wiley, 2005.
- [33] C. P. Robert, *Monte carlo methods*. Wiley Online Library, 2004.
- [34] C. Robert and G. Casella, *Monte Carlo statistical methods*. Springer Science & Business Media, 2013.
- [35] A. A. Shvets, A. Rakhlin, A. A. Kalinin, and V. I. Iglovikov, “Automatic instrument segmentation in robot-assisted surgery using deep learning,” in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 624–628, IEEE, 2018.
- [36] I. Guyon and A. Elisseeff, “An introduction to feature extraction,” in *Feature extraction*, pp. 1–25, Springer, 2006.
- [37] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh, *Feature extraction: foundations and applications*, vol. 207. Springer, 2008.

- [38] R. Kohavi, G. H. John, *et al.*, “Wrappers for feature subset selection,” *Artificial intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.
- [39] D. M. Witten and R. Tibshirani, “Penalized classification using fisher’s linear discriminant,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 73, no. 5, pp. 753–772, 2011.
- [40] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, vol. 1. Springer series in statistics New York, 2001.
- [41] N. Rachburee and W. Punlumjeak, “A comparison of feature selection approach between greedy, ig-ratio, chi-square, and mrmr in educational mining,” in *2015 7th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pp. 420–424, 2015.
- [42] L. Yu and H. Liu, “Feature selection for high-dimensional data: A fast correlation-based filter solution,” in *Proceedings of the 20th international conference on machine learning (ICML-03)*, pp. 856–863, 2003.
- [43] D. Mishra and B. Sahu, “Feature selection for cancer classification: a signal-to-noise ratio approach,” *International Journal of Scientific & Engineering Research*, vol. 2, no. 4, pp. 1–7, 2011.
- [44] R. O. Duda, P. E. Hart, and D. G. Stork, “Pattern classification. john wiley & sons,” *Inc., New York*, vol. 2, 2001.
- [45] E. Ghiselli, “Theory of psychological measurement. 1964,” 1964.
- [46] W. H. Press, S. A. Teukolsky, B. P. Flannery, and W. T. Vetterling, *Numerical recipes in Fortran 77: volume 1, volume 1 of Fortran numerical recipes: the art of scientific computing*. Cambridge university press, 1992.

- [47] M. Robnik-Šikonja and I. Kononenko, “Theoretical and empirical analysis of relieff and rrelieff,” *Machine learning*, vol. 53, no. 1-2, pp. 23–69, 2003.
- [48] H. Peng, F. Long, and C. Ding, “Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.
- [49] S. Visalakshi and V. Radha, “A literature review of feature selection techniques and applications: Review of feature selection in data mining,” in *2014 IEEE International Conference on Computational Intelligence and Computing Research*, pp. 1–6, IEEE, 2014.
- [50] T. Marill and D. Green, “On the effectiveness of receptors in recognition systems,” *IEEE transactions on Information Theory*, vol. 9, no. 1, pp. 11–17, 1963.
- [51] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping, “Use of the zero norm with linear models and kernel methods,” *The Journal of Machine Learning Research*, vol. 3, pp. 1439–1461, 2003.
- [52] A. Mojsilovic, “A logistic regression model for small sample classification problems with hidden variables and non-linear relationships: an application in business analytics,” in *Proceedings.(ICASSP’05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, vol. 5, pp. v–329, IEEE, 2005.
- [53] S. Ahmad, N. M. Ramli, and H. Midi, “Outlier detection in logistic regression and its application in medical data analysis,” in *2012 IEEE Colloquium*

- on Humanities, Science and Engineering (CHUSER)*, pp. 503–507, IEEE, 2012.
- [54] R. S. Collica, “A logistic regression yield model for sram bit fail patterns,” in *Proceedings of 1993 IEEE International Workshop on Defect and Fault Tolerance in VLSI Systems*, pp. 127–135, IEEE, 1993.
- [55] A. El-Koka, K.-H. Cha, and D.-K. Kang, “Regularization parameter tuning optimization approach in logistic regression,” in *2013 15th International Conference on Advanced Communications Technology (ICACT)*, pp. 13–18, IEEE, 2013.
- [56] A. Mojsilovic, “A logistic regression model for small sample classification problems with hidden variables and non-linear relationships: an application in business analytics,” in *Proceedings.(ICASSP’05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, vol. 5, pp. v–329, IEEE, 2005.
- [57] S. Ahmad, N. M. Ramli, and H. Midi, “Outlier detection in logistic regression and its application in medical data analysis,” in *2012 IEEE Colloquium on Humanities, Science and Engineering (CHUSER)*, pp. 503–507, IEEE, 2012.
- [58] R. S. Collica, “A logistic regression yield model for sram bit fail patterns,” in *Proceedings of 1993 IEEE International Workshop on Defect and Fault Tolerance in VLSI Systems*, pp. 127–135, IEEE, 1993.
- [59] D. T. Larose, “Logistic regression,” 2006.
- [60] A. Ng, “Cs229 lecture notes, part v: Support vector machines,” *Technical report*, 2012.

- [61] T. P. Minka, “A comparison of numerical optimizers for logistic regression,” *Unpublished draft*, pp. 1–18, 2003.
- [62] S.-I. Lee, H. Lee, P. Abbeel, and A. Y. Ng, “Efficient  $l_1$  regularized logistic regression,” in *Aaai*, vol. 6, pp. 401–408, 2006.
- [63] S. Koleva and J. Haidt, “Let’s use einstein’s safety razor, not occam’s swiss army knife or occam’s chainsaw,” *Psychological Inquiry*, vol. 23, no. 2, pp. 175–178, 2012.
- [64] X. Li, “Finding deterministic solution from underdetermined equation: large-scale performance variability modeling of analog/rf circuits,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 11, pp. 1661–1668, 2010.
- [65] S. Perkins and J. Theiler, “Online feature selection using grafting,” in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 592–599, 2003.
- [66] H. Akaike, “A new look at the statistical model identification,” *IEEE transactions on automatic control*, vol. 19, no. 6, pp. 716–723, 1974.
- [67] C. L. Mallows, “Some comments on  $cp$ ,” *Technometrics*, vol. 42, no. 1, pp. 87–94, 2000.
- [68] M. Kwemou, M.-L. Taupin, and A.-S. Tocquet, “Model selection in logistic regression,” *arXiv preprint arXiv:1508.07537*, 2015.
- [69] S. Olejnik, J. Mills, and H. Keselman, “Using wherry’s adjusted  $r^2$  and mallow’s  $cp$  for model selection from all possible regressions,” *The Journal of experimental education*, vol. 68, no. 4, pp. 365–380, 2000.

- [70] H. Akaike, “Information theory and an extension of the maximum likelihood principle,” in *Selected papers of hirotugu akaike*, pp. 199–213, Springer, 1998.
- [71] C. M. Hurvich and C.-L. Tsai, “Regression and time series model selection in small samples,” *Biometrika*, vol. 76, no. 2, pp. 297–307, 1989.
- [72] K. P. Burnham and D. R. Anderson, “A practical information-theoretic approach,” *Model selection and multimodel inference*, vol. 2, 2002.
- [73] H. Yanagihara, R. Sekiguchi, and Y. Fujikoshi, “Bias correction of aic in logistic regression models,” *Journal of statistical planning and inference*, vol. 115, no. 2, pp. 349–360, 2003.
- [74] M. B. Alawieh, F. Wang, R. Kanj, X. Li, and R. Joshi, “Efficient analog circuit optimization using sparse regression and error margining,” in *Intl. Symposium on Quality Electronic Design*, pp. 410–415, IEEE, 2016.
- [75] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani, *et al.*, “Least angle regression,” *Annals of statistics*, vol. 32, no. 2, pp. 407–499, 2004.
- [76] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.
- [77] M. Y. Park and T. Hastie, *Regularization path algorithms for detecting gene interactions*. Department of Statistics, Stanford University California, USA, 2006.
- [78] M. Yuan and Y. Lin, “Model selection and estimation in regression with grouped variables,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 1, pp. 49–67, 2006.

- [79] L. Demidova and I. Klyueva, “Improving the classification quality of the svm classifier for the imbalanced datasets on the base of ideas the smote algorithm,” in *ITM Web of Conferences*, vol. 10, p. 02002, EDP Sciences, 2017.
- [80] M. Schubach, M. Re, P. N. Robinson, and G. Valentini, “Imbalance-aware machine learning for predicting rare and common disease-associated non-coding variants,” *Scientific reports*, vol. 7, no. 1, pp. 1–12, 2017.
- [81] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, “An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics,” *Information sciences*, vol. 250, pp. 113–141, 2013.
- [82] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, “A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 4, pp. 463–484, 2011.
- [83] M. V. Joshi, V. Kumar, and R. C. Agarwal, “Evaluating boosting algorithms to classify rare classes: Comparison and improvements,” in *Proceedings 2001 IEEE International Conference on Data Mining*, pp. 257–264, IEEE, 2001.
- [84] G. M. Weiss and F. Provost, “Learning when training data are costly: The effect of class distribution on tree induction,” *Journal of artificial intelligence research*, vol. 19, pp. 315–354, 2003.

- [85] N. Japkowicz and S. Stephen, “The class imbalance problem: A systematic study,” *Intelligent data analysis*, vol. 6, no. 5, pp. 429–449, 2002.
- [86] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.
- [87] H. Han, W.-Y. Wang, and B.-H. Mao, “Borderline-smote: a new over-sampling method in imbalanced data sets learning,” in *International conference on intelligent computing*, pp. 878–887, Springer, 2005.
- [88] I. Mani and I. Zhang, “knn approach to unbalanced data distributions: a case study involving information extraction,” in *Proceedings of workshop on learning from imbalanced datasets*, vol. 126, ICML United States, 2003.
- [89] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.
- [90] M. Malik, R. V. Joshi, R. Kanj, S. Sun, H. Homayoun, and T. Li, “Sparse regression driven mixture importance sampling for memory design,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 1, pp. 63–72, 2018.
- [91] T. Lee, K. B. Lee, and C. O. Kim, “Performance of machine learning algorithms for class-imbalanced process fault detection problems,” *IEEE Transactions on Semiconductor Manufacturing*, vol. 29, no. 4, pp. 436–445, 2016.
- [92] P. Jeatrakul, K. W. Wong, and C. C. Fung, “Classification of imbalanced data by combining the complementary neural network and smote

- algorithm,” in *International Conference on Neural Information Processing*, pp. 152–159, Springer, 2010.
- [93] R. Dietrich, M. Opper, and H. Sompolinsky, “Statistical mechanics of support vector networks,” *Physical review letters*, vol. 82, no. 14, p. 2975, 1999.
- [94] H. He and Y. Ma, “Imbalanced learning: foundations, algorithms, and applications,” 2013.
- [95] G. Wu and E. Y. Chang, “Adaptive feature-space conformal transformation for imbalanced-data learning,” in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 816–823, 2003.
- [96] G. Wu and E. Y. Chang, “Class-boundary alignment for imbalanced dataset learning,” in *ICML 2003 workshop on learning from imbalanced data sets II, Washington, DC*, pp. 49–56, 2003.
- [97] R. Akbani, S. Kwek, and N. Japkowicz, “Applying support vector machines to imbalanced datasets,” in *European conference on machine learning*, pp. 39–50, Springer, 2004.
- [98] G. Wu and E. Y. Chang, “Class-boundary alignment for imbalanced dataset learning,” in *ICML 2003 workshop on learning from imbalanced data sets II, Washington, DC*, pp. 49–56, 2003.
- [99] H. He and E. A. Garcia, “Learning from imbalanced data,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

- [100] I. Mani and I. Zhang, “knn approach to unbalanced data distributions: a case study involving information extraction,” in *Proceedings of workshop on learning from imbalanced datasets*, vol. 126, 2003.
- [101] R. Batuwita and V. Palade, “Efficient resampling methods for training support vector machines with imbalanced datasets,” in *2010 International Joint Conference on Neural Networks*, pp. 1–8, IEEE, 2010.
- [102] Z. Lin, Z. Hao, X. Yang, and X. Liu, “Several svm ensemble methods integrated with under-sampling for imbalanced data learning,” in *International conference on advanced data mining and applications*, pp. 536–544, Springer, 2009.
- [103] I. Albusua, O. Arbelaitz, I. Gurrutxaga, A. Lasarguren, J. Muguerza, and J. M. Pérez, “The quest for the optimal class distribution: an approach for enhancing the effectiveness of learning via resampling methods for imbalanced data sets,” *Progress in Artificial Intelligence*, vol. 2, no. 1, pp. 45–63, 2013.
- [104] A. Nath and K. Subbiah, “Probing an optimal class distribution for enhancing prediction and feature characterization of plant virus-encoded rna-silencing suppressors,” *3 Biotech*, vol. 6, no. 1, p. 93, 2016.
- [105] Y. Lin, Y. Lee, and G. Wahba, “Support vector machines for classification in nonstandard situations,” *Machine learning*, vol. 46, no. 1-3, pp. 191–202, 2002.
- [106] K. Veropoulos, C. Campbell, N. Cristianini, *et al.*, “Controlling the sensitivity of support vector machines,” in *Proceedings of the international joint conference on AI*, vol. 55, p. 60, 1999.

- [107] V. Palade, “Class imbalance learning methods for support vector machines,” *Imbalanced Learning: Foundations, Algorithms, and Applications*; Wiley: Hoboken, NJ, USA, p. 83, 2013.
- [108] N. V. Chawla, “C4. 5 and imbalanced data sets: investigating the effect of sampling method, probabilistic estimate, and decision tree structure,” in *Proceedings of the ICML*, vol. 3, p. 66, 2003.
- [109] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik, “Feature selection for svms,” in *Advances in neural information processing systems*, pp. 668–674, 2001.
- [110] S. Perkins, K. Lacker, and J. Theiler, “Grafting: Fast, incremental feature selection by gradient descent in function space,” *Journal of machine learning research*, vol. 3, no. Mar, pp. 1333–1356, 2003.
- [111] G. Forman, “An extensive empirical study of feature selection metrics for text classification,” *Journal of machine learning research*, vol. 3, no. Mar, pp. 1289–1305, 2003.
- [112] Z. Zheng, X. Wu, and R. Srihari, “Feature selection for text categorization on imbalanced data,” *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 80–89, 2004.
- [113] V. N. Vapnik, “The nature of statistical learning theory,” *New York: Springer-Verlag*, 1995.
- [114] O. Chapelle, P. Haffner, and V. N. Vapnik, “Support vector machines for histogram-based image classification,” *IEEE transactions on Neural Networks*, vol. 10, no. 5, pp. 1055–1064, 1999.

- [115] Z. Zhang, X. Gu, Y. Xie, Z. Wang, Z. Wang, and K. Chakrabarty, “Diagnostic system based on support-vector machines for board-level functional diagnosis,” in *2012 17th IEEE European Test Symposium (ETS)*, pp. 1–6, IEEE, 2012.
- [116] S. Maldonado, R. Weber, and J. Basak, “Simultaneous feature selection and classification using kernel-penalized support vector machines,” *Information Sciences*, vol. 181, no. 1, pp. 115–128, 2011.
- [117] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.
- [118] K. Veropoulos, C. Campbell, N. Cristianini, *et al.*, “Controlling the sensitivity of support vector machines,” in *Proceedings of the international joint conference on AI*, vol. 55, p. 60, 1999.
- [119] R. Akbani, S. Kwek, and N. Japkowicz, “Applying support vector machines to imbalanced datasets,” in *European conference on machine learning*, pp. 39–50, Springer, 2004.
- [120] T. Imam, K. M. Ting, and J. Kamruzzaman, “z-svm: An svm for improved classification of imbalanced data,” in *Australasian Joint Conference on Artificial Intelligence*, pp. 264–273, Springer, 2006.
- [121] N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor, “On kernel target alignment,” in *Innovations in machine learning*, pp. 205–256, Springer, 2006.

- [122] T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu, “The entire regularization path for the support vector machine,” *Journal of Machine Learning Research*, vol. 5, no. Oct, pp. 1391–1415, 2004.
- [123] J. Wang, X. Shen, and Y. Liu, “Probability estimation for large-margin classifiers,” *Biometrika*, vol. 95, no. 1, pp. 149–167, 2008.
- [124] M. Kuhn and K. Johnson, “An introduction to feature selection,” in *Applied predictive modeling*, pp. 487–519, Springer, 2013.
- [125] G. From, “Jump to: navigation, search (hastie et al., 2004) trevor hastie, saharon rosset, robert tibshirani, ji zhu.(2004).“the entire regularization path for the support vector machine.” in: The journal of machine learning research, 5. subject headings: Svm classification algorithm, regularization path, algorithm tuning parameter, regularization cost parameter, kernel function parameter, cost parameter.”
- [126] J. Zhu, S. Rosset, R. Tibshirani, and T. J. Hastie, “1-norm support vector machines,” in *Advances in neural information processing systems*, p. None, Citeseer, 2003.
- [127] J. Bi, K. Bennett, M. Embrechts, C. Breneman, and M. Song, “Dimensionality reduction via sparse support vector machines,” *Journal of Machine Learning Research*, vol. 3, no. Mar, pp. 1229–1243, 2003.
- [128] S. S. Keerthi, “Generalized lars as an effective feature selection tool for text classification with svms,” in *Proceedings of the 22nd international conference on Machine learning*, pp. 417–424, 2005.

- [129] A. B. Chan, N. Vasconcelos, and G. R. Lanckriet, “Direct convex relaxations of sparse svm,” in *Proceedings of the 24th international conference on Machine learning*, pp. 145–153, 2007.
- [130] Y. Yang and H. Zou, “An efficient algorithm for computing the hhsvm and its generalizations,” *Journal of Computational and Graphical Statistics*, vol. 22, no. 2, pp. 396–415, 2013.
- [131] C. Yi and Y. Zeng, *sparseSVM: Solution Paths of Sparse High-Dimensional Support Vector Machine with Lasso or Elastic-Net Regularization*, 2018. R package version 1.1-6.
- [132] B. Efron and R. J. Tibshirani, *Cross-validation and the bootstrap: Estimating the error rate of a prediction rule*. Division of Biostatistics, Stanford University, 1995.
- [133] S. Rosset and J. Zhu, “Piecewise linear regularized solution paths,” *The Annals of Statistics*, pp. 1012–1030, 2007.
- [134] L. Wang, J. Zhu, and H. Zou, “Hybrid huberized support vector machines for microarray classification and gene selection,” *Bioinformatics*, vol. 24, no. 3, pp. 412–419, 2008.
- [135] Y. Zhou, M. Han, L. Liu, J. S. He, and Y. Wang, “Deep learning approach for cyberattack detection,” in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 262–267, IEEE, 2018.
- [136] G. King and L. Zeng, “Logistic regression in rare events data,” *Political analysis*, vol. 9, no. 2, pp. 137–163, 2001.

- [137] N. Japkowicz and S. Stephen, “The class imbalance problem: A systematic study,” *Intelligent data analysis*, vol. 6, no. 5, pp. 429–449, 2002.
- [138] G. M. Weiss and F. Provost, “Learning when training data are costly: The effect of class distribution on tree induction,” *Journal of artificial intelligence research*, vol. 19, pp. 315–354, 2003.
- [139] R. Joshi, R. Kanj, S. Nassif, D. Plass, Y. Chan, *et al.*, “Statistical exploration of the dual supply voltage space of a 65nm pd/soi cmos sram cell,” in *2006 European Solid-State Device Research Conference*, pp. 315–318, IEEE, 2006.
- [140] G. Yu and P. Li, “Yield-aware hierarchical optimization of large analog integrated circuits,” in *2008 IEEE/ACM International Conference on Computer-Aided Design*, pp. 79–84, IEEE, 2008.
- [141] R. Kanj, R. Joshi, and S. Nassif, “Mixture importance sampling and its application to the analysis of sram designs in the presence of rare failure events,” in *2006 43rd ACM/IEEE Design Automation Conference*, pp. 69–72, IEEE, 2006.
- [142] M. V. Joshi, *Learning classifier models for predicting rare phenomena*. University of Minnesota, 2002.
- [143] G. Lemaître, F. Nogueira, and C. K. Aridas, “Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning,” *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 559–563, 2017.

- [144] R. V. Joshi and M. M. Ziegler, “Programmable supply boosting techniques for near threshold and wide operating voltage sram,” in *2017 IEEE Custom Integrated Circuits Conference*, pp. 1–4, IEEE, 2017.
- [145] S. J. Shin, *wsvmsurf: Two-dimensional Solution Surface of the Weighted Support Vector Machine algorithm*, 2010. R package version 1.0.
- [146] N. Rout, D. Mishra, and M. K. Mallick, “Handling imbalanced data: a survey,” in *International Proceedings on Advances in Soft Computing, Intelligent Systems and Applications*, pp. 431–443, Springer, 2018.
- [147] Y. Chen, Y. Lin, T. Gai, Y. Su, Y. Wei, and D. Z. Pan, “Semisupervised hotspot detection with self-paced multitask learning,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 7, pp. 1511–1523, 2019.