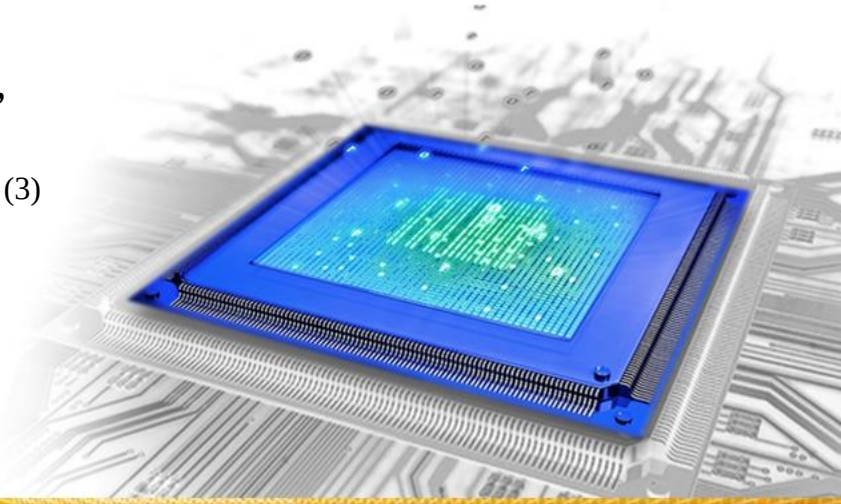
Timing Assistant for Dynamic Voltage Drop Impact on Maximum Timing Pushout

Norman Chang [1], Wentze Chuang [1,3] , Ganesh Kumar Tsavatapalli [1],
Hao Zhuang [2], Sankar Ramachandran [1], Rahul Rajan [1], Joao Geada [1],
Ying-Shiun Li [1], Yaowei Jia [1], Mathew Kaipanatu [1],
Suresh Kumar Mantena [1], Ming-Chih Shih [1], Anita Yang [1], Roger Jang [3]

[1] ANSYS Inc., [2] Google Inc., [3] National Taiwan University



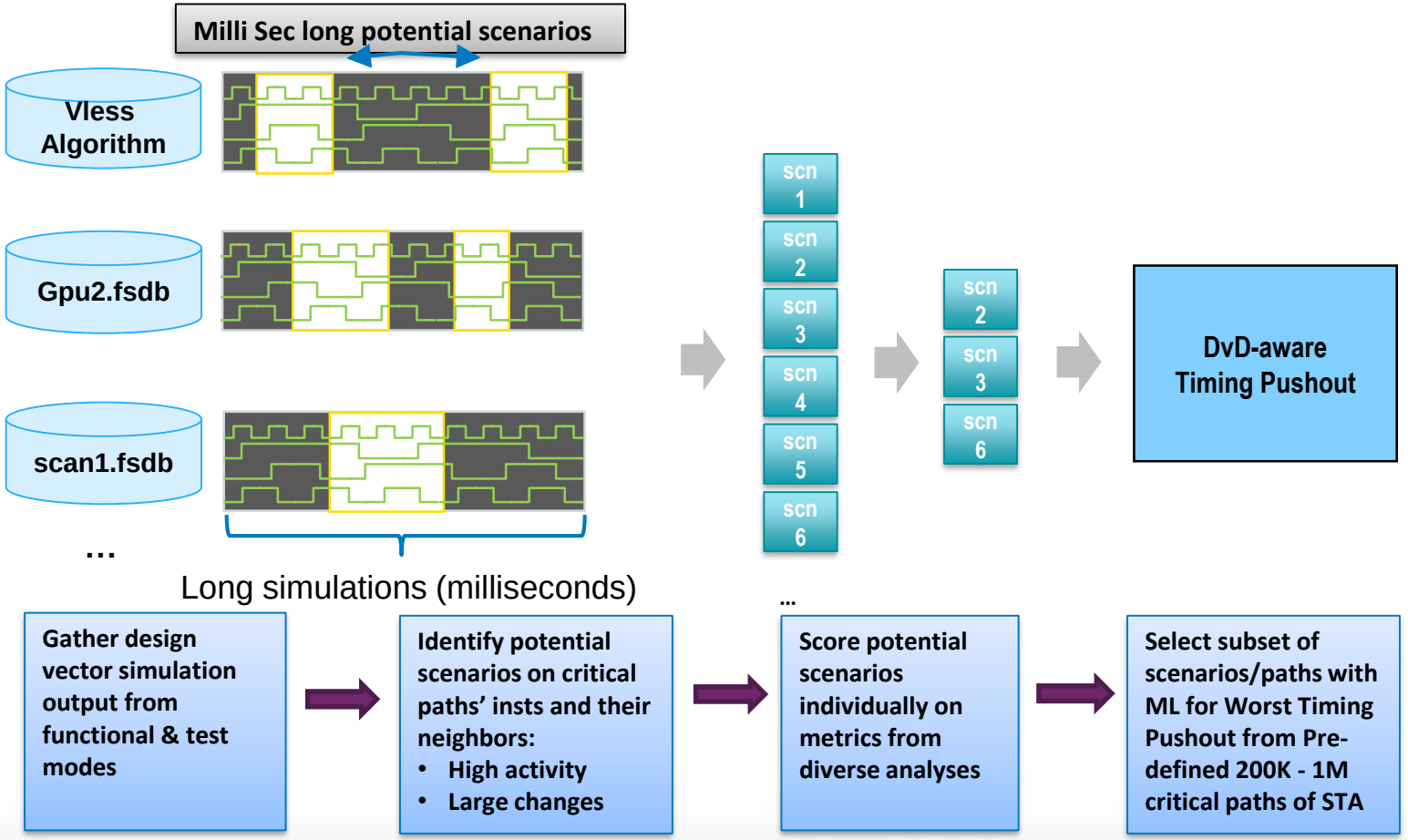
Outline

- Introduction
- Timing Assistant Methodology and Flow
- Results and Discussion
- Conclusions

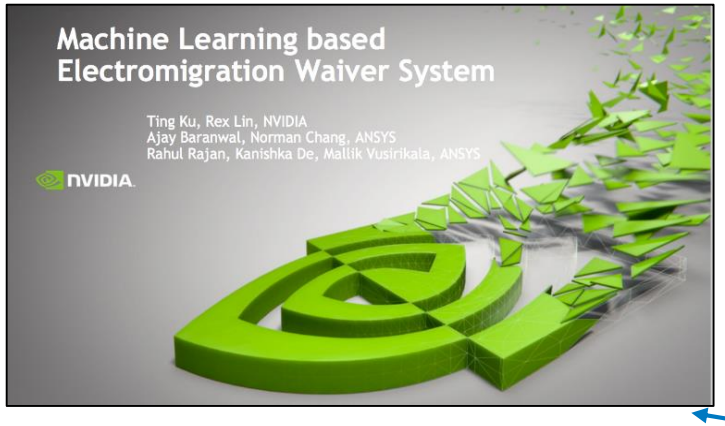
Scenario Selection and Timing Path Selection for DvD-aware Maximum Timing Pushout

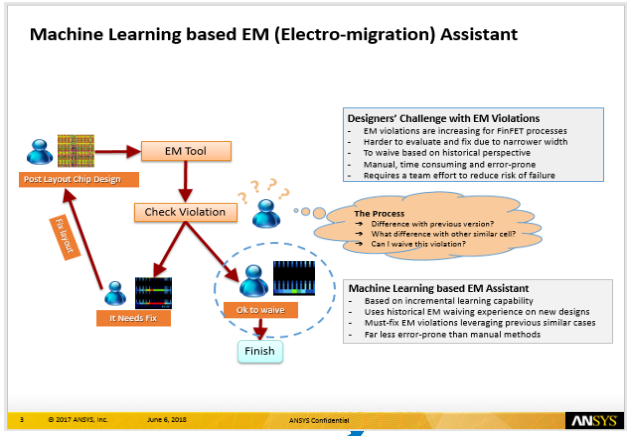
- The flow creates maximum timing pushout scenarios/paths from multiple scenarios of both vector and vectorless algorithms based on various parameters from diverse analyses.

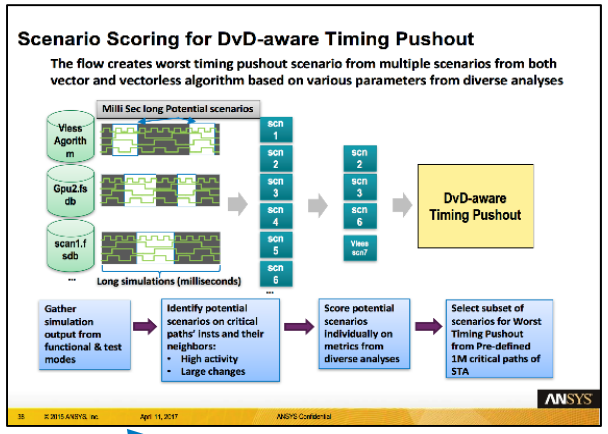
- This process can be assisted with ML-based Timing Assistant to learn from historical data for more accurate selection of scenarios and paths that may produce maximum timing pushout.

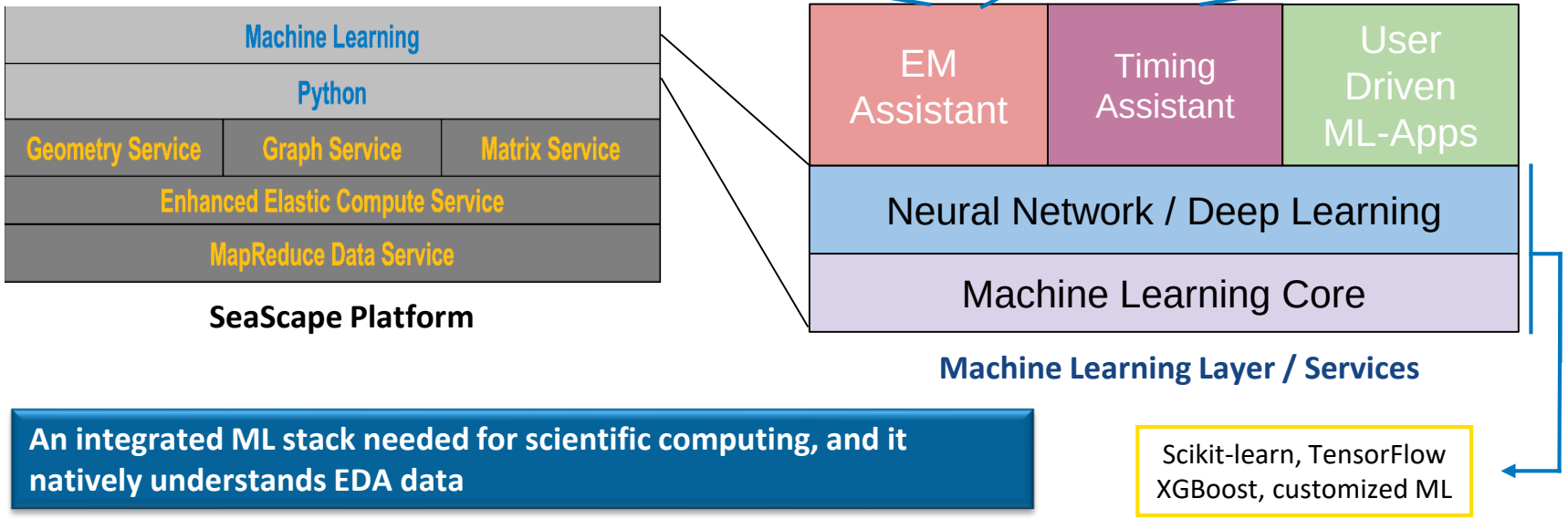


ML/DL : ML-SC EM/Timing Assistant for Reliability/Timing Check





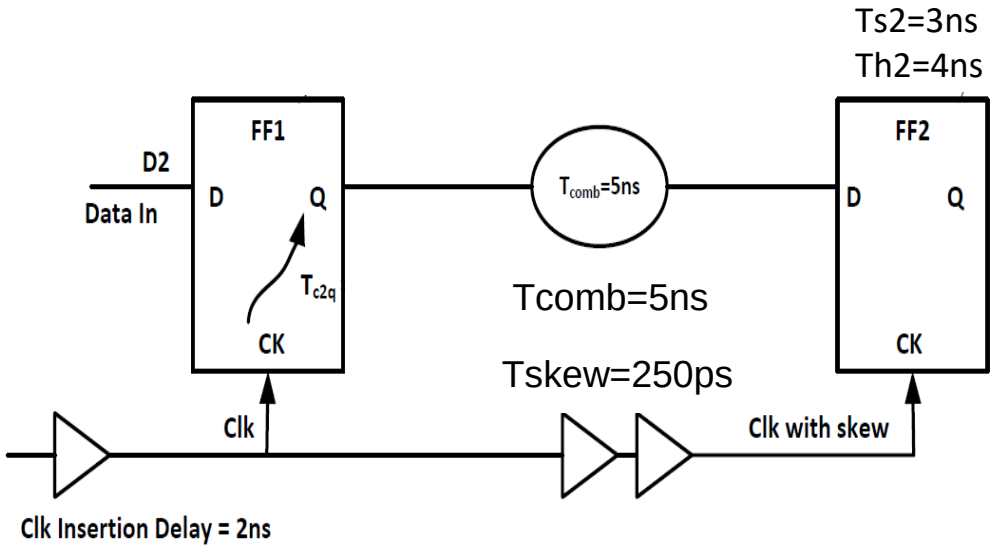




An integrated ML stack needed for scientific computing, and it natively understands EDA data

Example Applications: "Machine Learning based Generic Violation Waiver System with Application on Electromigration Sign-off, N. Chang, T. Ku (Nvidia), et al, ASP-DAC, 2018. Applying Machine Learning to Design for Reliability Coverage, N. Chang, et al., IRPS 2019

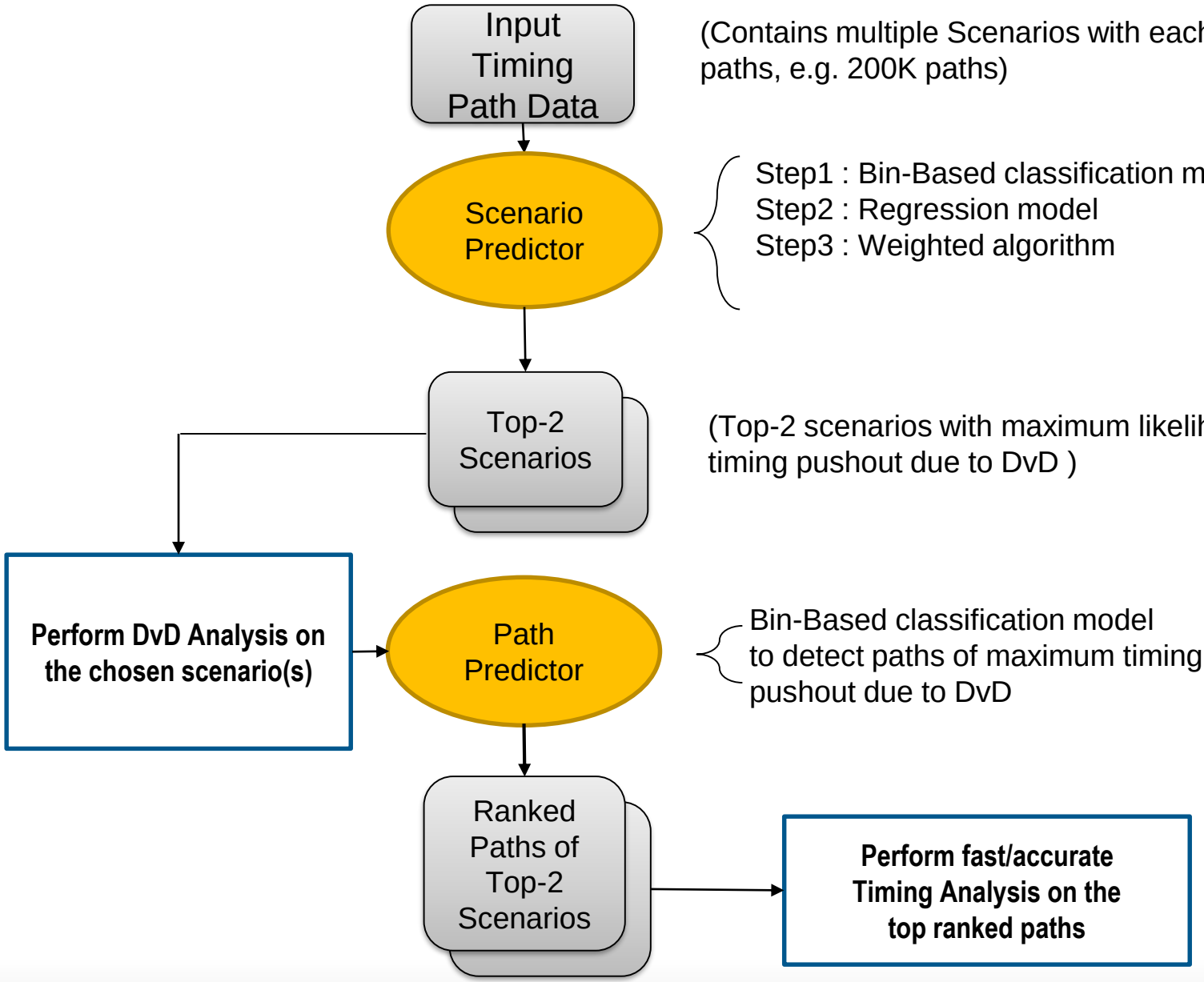
Definition of Setup Time Delta Slack due to DvD



Ts2 = setup time needed of FF2
Th2 = hold time needed of FF2
Tc2q = Time from clock pin to Q pin of FF
Tskew = Time for the clock path
Tcomb = Time for the combinational instances on the Data Path

- **Tc2q and Tcomb will be larger when considering dynamic voltage drop (DvD) of instances along the Data Path.**
  The corresponding eq. for maximum operation frequency is defined below:
  Fmax = 1 / (Tclk) where Tclk = Tc2q + Tcomb + Ts2 – Tskew
- Data Arrival Time : Time of data arriving input-D of FF2
- Data Required Time : The latest time at which a signal can arrive without making the clock cycle longer than desired.
- Setup Slack : Data Required Time – Data Arrival Time
- Pre DvD Slack : The Slack value without considering DvD
- Post DvD Slack : The Slack value when considering DvD on data path
- **Delta Slack : Post DvD Slack – Pre DvD Slack**

System Flow of Scenario Predictor and Path Predictor



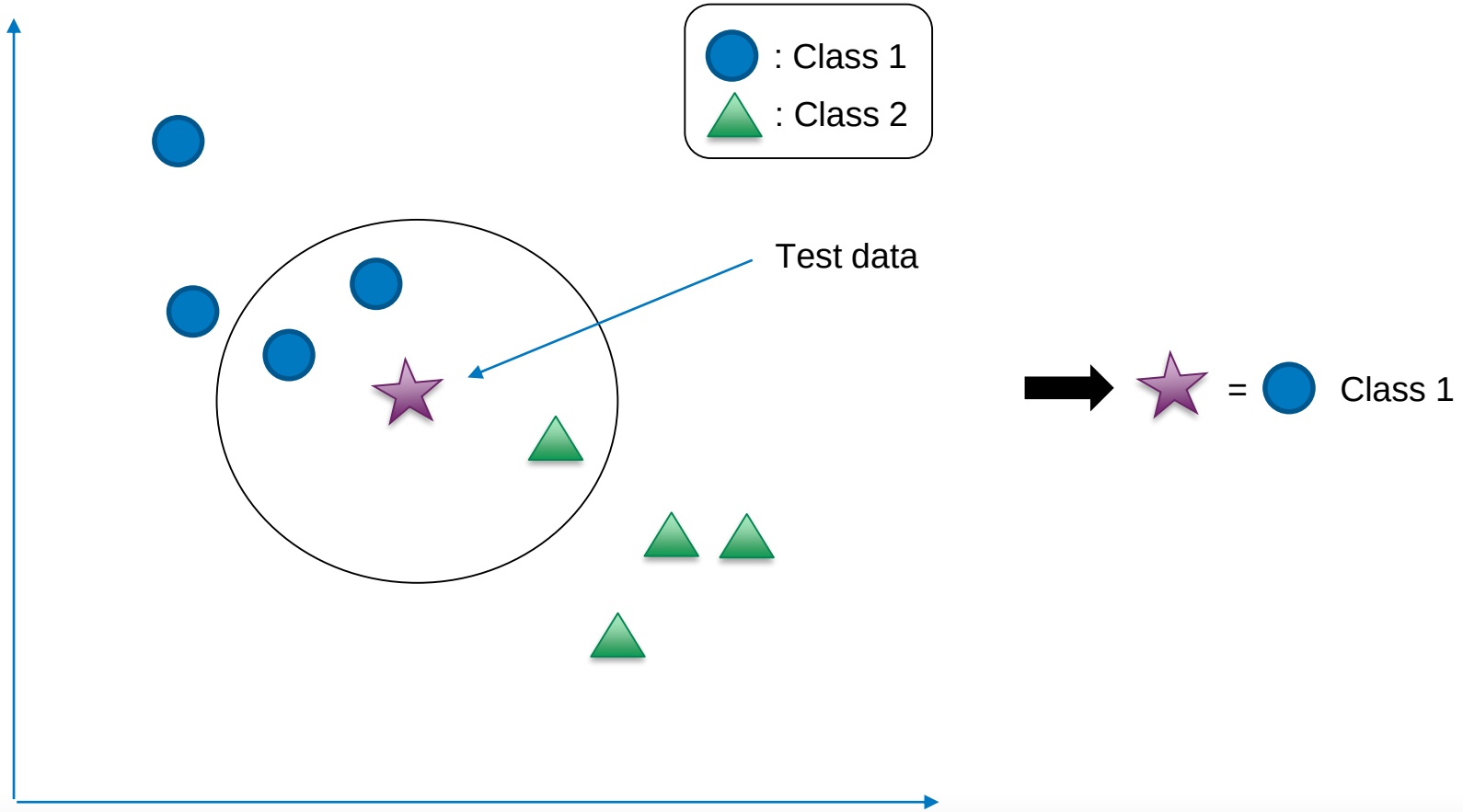
ML Algorithm Applied

- KNN (K nearest neighbors)
- Decision Tree
- Random Forest
- XGBoost (eXtreme gradient boosting)
- lambdaMART (learning to rank technique)
- Neural Networks

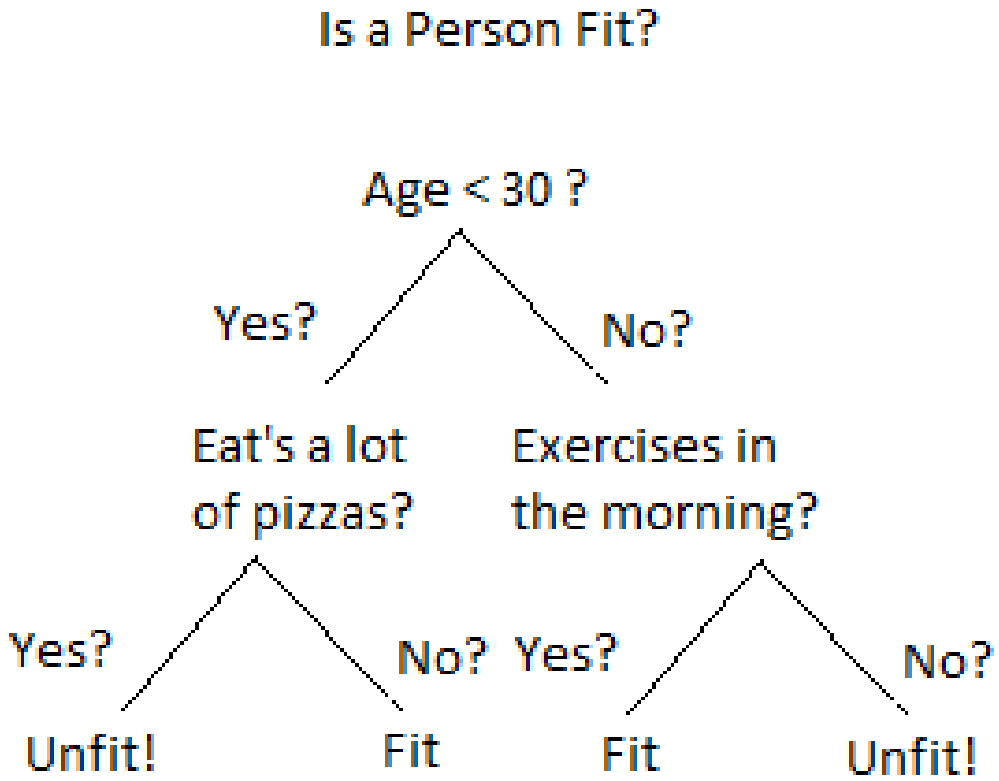
ML Algorithm Applied – KNN (K-nearest Neighbors) Algorithm

- Test data will be classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors

- Example , K = 3 :



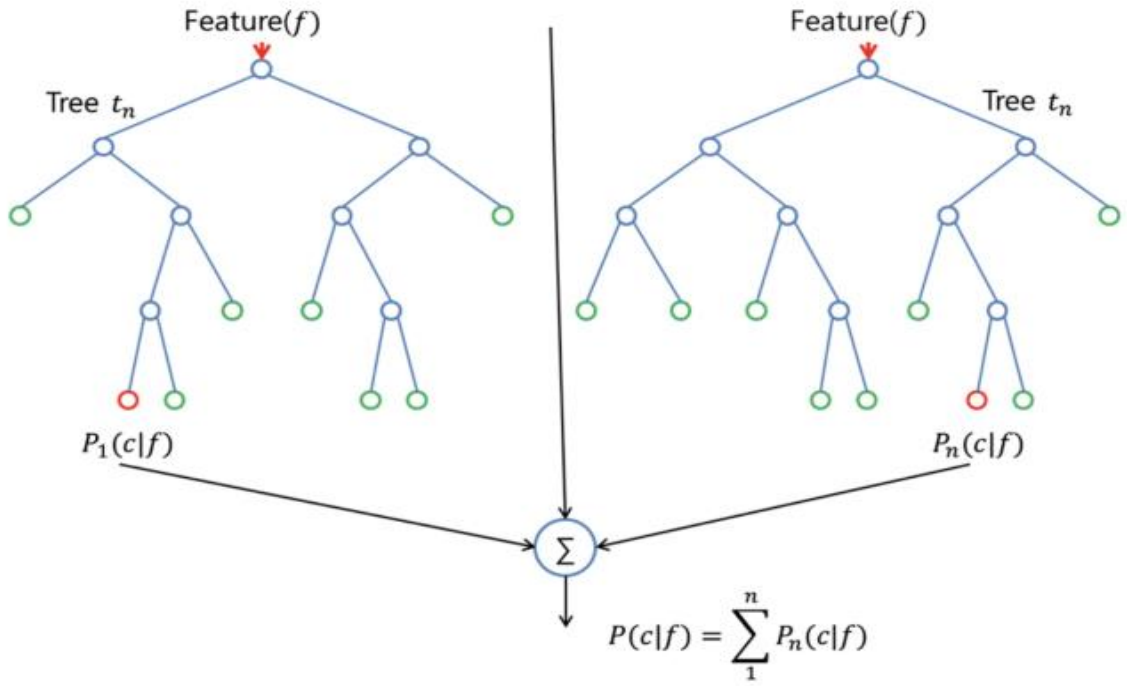
ML Algorithm Applied – Decision Tree Algorithm



James, Gareth; Witten, Daniela; Hastie, Trevor; Tibshirani, Robert (2017). "Tree-Based Methods". An Introduction to Statistical Learning: with Applications in R. New York: Springer. pp. 303–336. ISBN 978-1-4614-7137-0.

ML Algorithm Applied – Random Forests Algorithm

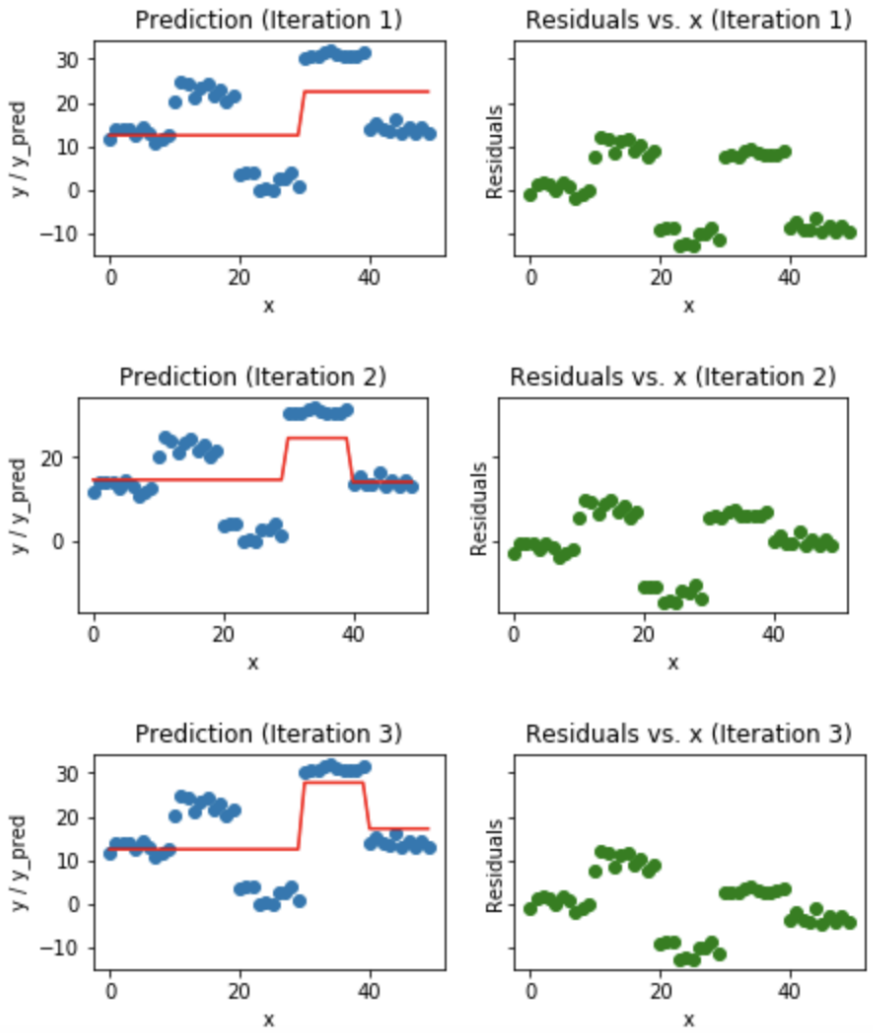
- Train classifiers based on a subset of features
- Aggregate output of trees (majority vote or averaging) as a typical ensemble learning method



L. Breiman, "Random Forests." *Machine learning* 45.1 (2001): 5-32

XGBoost (Extreme Gradient Boosting)

- XGBoost is an effective gradient-based boosting method that iteratively improve the classifier starting with weak classifiers
- Source code available (not in Scikit-learn yet but open source); ported to SeaScape
  - Supports multiple languages and platforms
  - Distributed on various clouds
  - Wins many ML competitions
  - Highly scalable and tunable



$$y_i^p = y_i^p + \alpha * \delta \sum (y_i - y_i^p)^2 / \delta y_i^p$$

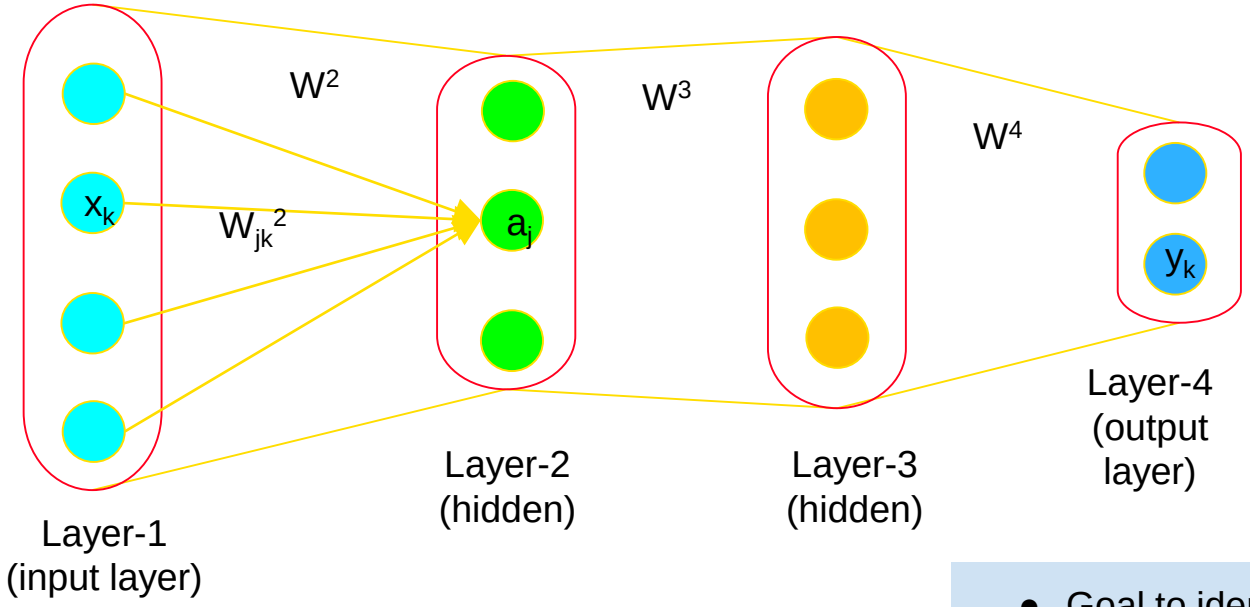
which becomes, $y_i^p = y_i^p - \alpha * 2 * \sum (y_i - y_i^p)$

where, α is learning rate and $\sum (y_i - y_i^p)$ is sum of residuals

T. Chen, C. Guestrin, "A Scalable Tree Boosting System", KDD 2016.

In 2015, 17 out of 29 Kaggle champions used XGBoost in their solutions.

Neural Networks



- x ⇒ Input feature vector
- W^l ⇒ Weights matrix b/w (l-1) to l^{th} layer
  - W^l_{jk}⇒ weight from k^{th} neuron from $(l-1)^{th}$ layer to j^{th} neuron in l^{th} layer
- b^l ⇒ Bias matrix of l^{th} layer
- z^l ⇒ weighted input of l^{th} layer
- a^l ⇒ Activation matrix of l^{th} layer
- y ⇒ Output label
- C ⇒ Total Cost (E.g. Square Sum Error)
- d^l ⇒ Error at l^{th} layer
- D^l ⇒ dC/dW ⇒ Partial derivative of cost w.r.t weights

- Goal to identify optimal weights (W's) and bias (b's) for modeling regression or classification
- After having partial derivatives at every layer, we can train model using a learning rate and stochastic gradient decent methods.

Schmidhuber, J. (2015). "Deep Learning in Neural Networks: An Overview". *Neural Networks*. **61**: 85–117. arXiv:1404.7828

LambdaMART

- LambdaMART is the boosted tree version of LambdaRank, which is based on RankNet. RankNet, LambdaRank, and LambdaMART have proven to be very successful algorithms for solving real world ranking problems: for example an ensemble of LambdaMART rankers won Track 1 of the 2010 Yahoo! Learning To Rank Challenge.

- Solving a wide range of supervised learning problem by maximizing information retrieval (IR) functions, like NDCG, which are not smooth functions of the model scores

C. Burges “From RankNet to LambdaRank to LambdaMART : an Overview”, Microsoft Research Technical Report MSR-TR-2010-82

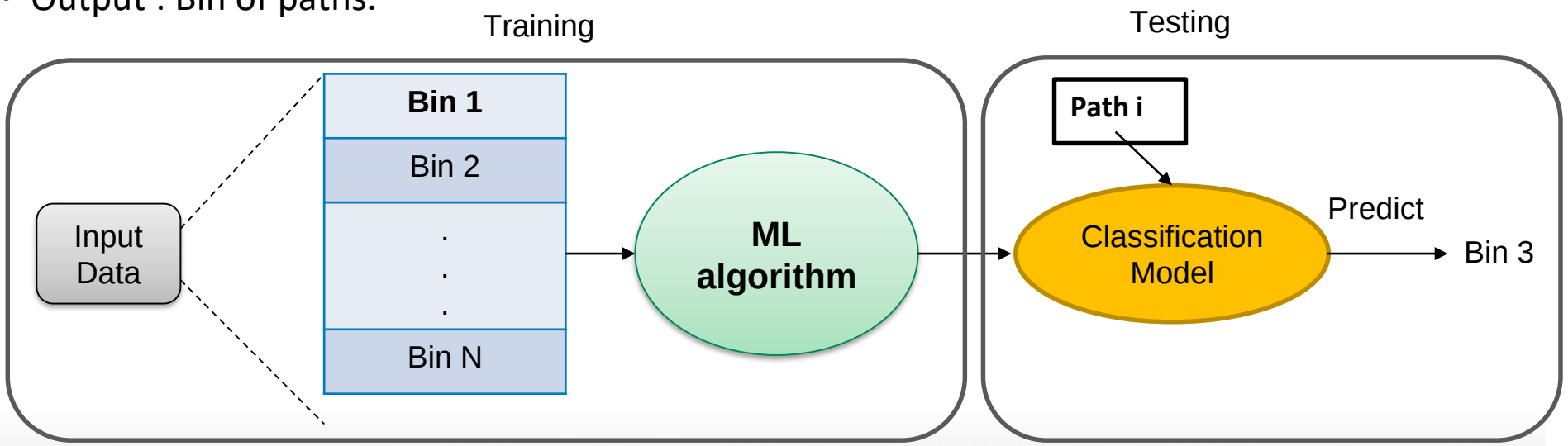
Possible Path-based Input Features Including Customized Combined Features

- Avg_timing_sensitivity_wrt_dvd : avg of per instance timing sensitivity w.r.t. DvD on the Data Path
- Max_timing_sensitivity_wrt_dvd : max of per instance timing sensitivity w.r.t. DvD on the Data Path
- Avg_switching_status : avg of per instance switching state (1 or 0 per instance switching)
- Avg_effective_R : avg of per instance effective R from bump to instance VDD/VSS pins
- Max_effective_R : max of per instance effective R from bump to instance VDD/VSS pins
- Num_of_datapath_instances : number of instances on the Data Path
- Avg_rail_connected_datapath_Ipeak : avg of rail-connected peak current per instance on the Data Path
- Max_rail_connected_datapath_Ipeak : max of rail-connected peak current per instance on the Data Path
- Avg_effective_decap : avg of effective decap seen by each instance on the Data Path
- Max_effective_decap : max of effective decap seen by instances on the Data Path
- Package_effect : package impact on Data Path instances delay. e.g. Ldi/dt
- Combined_timing_impact_features : e.g., Avg_rail_connected_Ipeak * Avg_effective_R * Avg_timing_sensitivity_wrt_dvd * Num_of datapath_instances (i.e. from feature engineering)
- Avg_worst_voltage_drop : avg of per instance worst voltage drop on the Data Path (additional for Path Predictor)
- Max_worst_voltage_drop : max of per instance worst voltage drop on the Data Path (additional for Path Predictor)

.
.
.

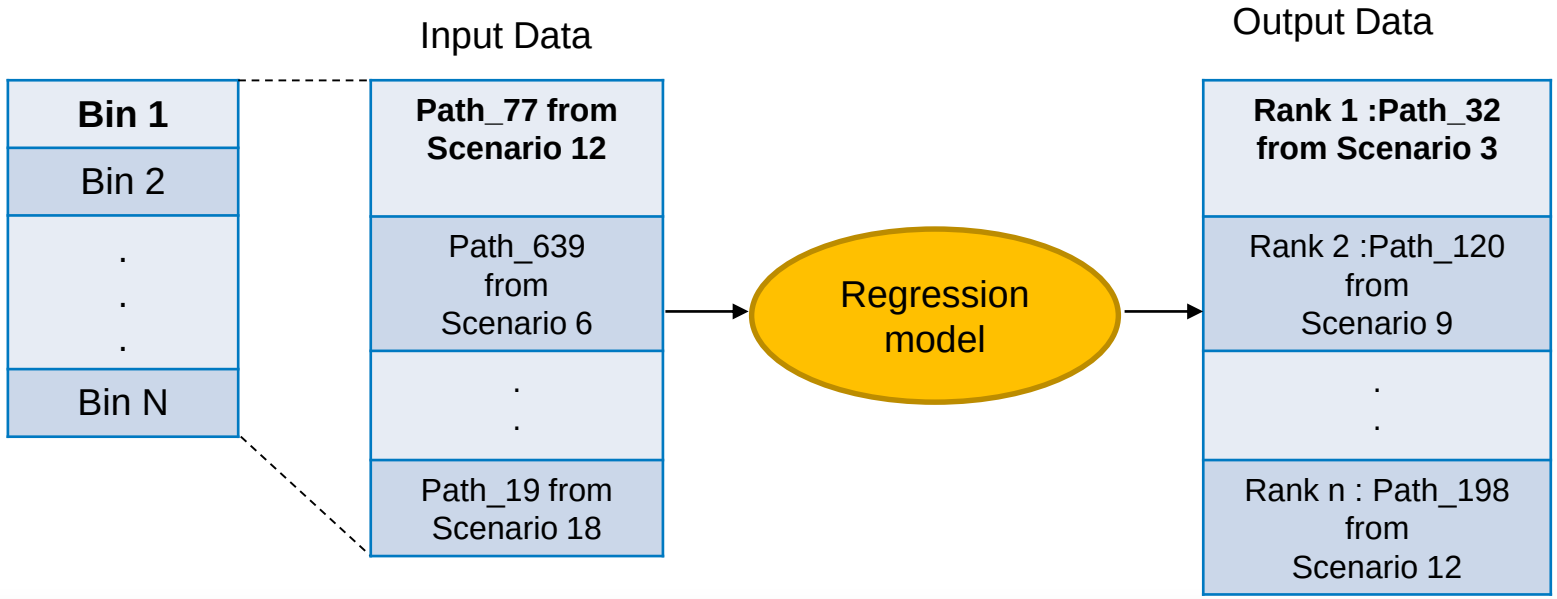
Scenario Predictor : Step 1 - Bin-based Classification Model

- Input data : 20 Scenarios, 70K timing paths per scenario for training and 30K for testing.
- Split Input data into N bins according to delta slack.
- ML algorithm:
  - KNN
  - Decision tree and Random Forest
  - XGBoost
  - NN
- Input Feature : 17 features
- Output : Bin of paths.



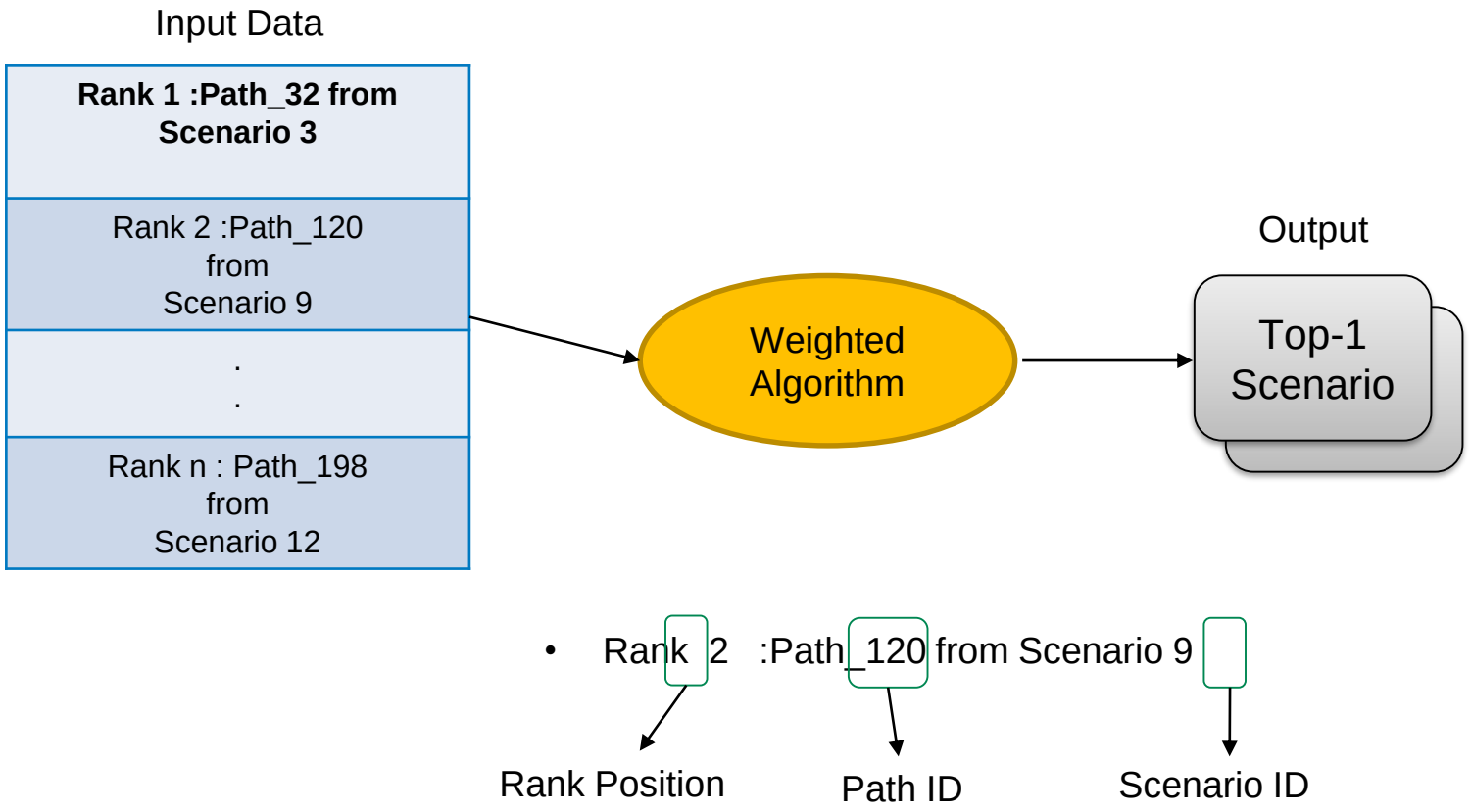
Scenario Predictor : Step 2 – Ranking Model

- Input Data : Output of step 1 (Top-bin data)
- ML algorithm:
  - Decision Tree regressor
  - XGBoost regressor
  - lambdaMART (learning to rank technique)
- Input Feature : 17 features
- Output : Ranked Timing Paths based on Predicted Delta Slack from Regression model (Rank 1: The **most negative** delta slack).



Scenario Predictor : Step 3 – Weighted Algorithm

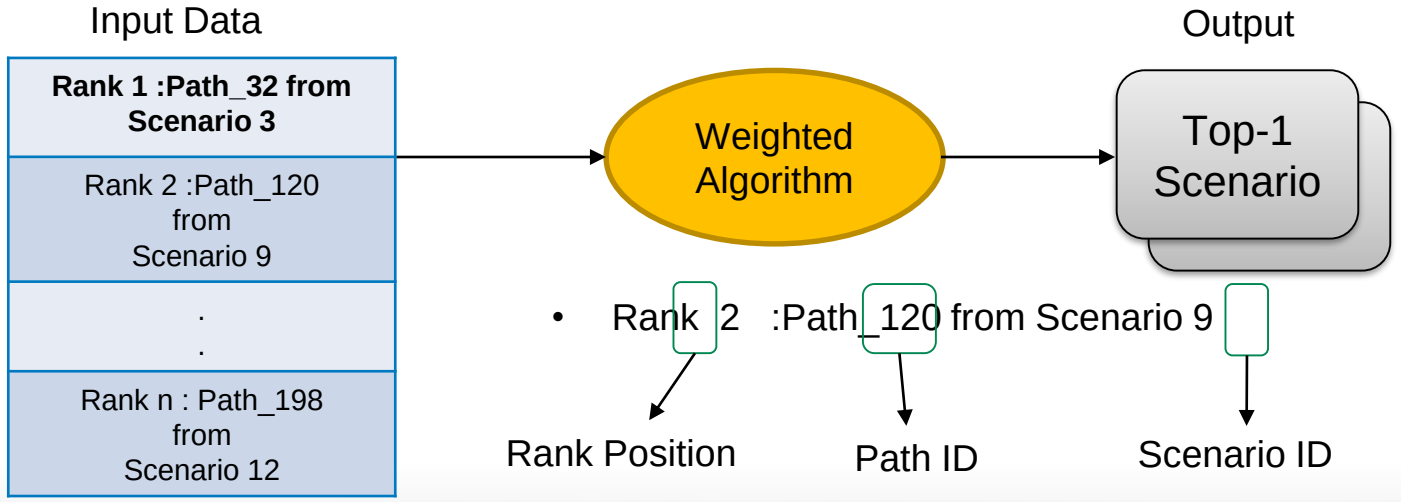
- Input Data : Output of Regression model (step2)
- Output : Top-K scenarios (e.g., K=2)



System Details – Weighted Algorithm

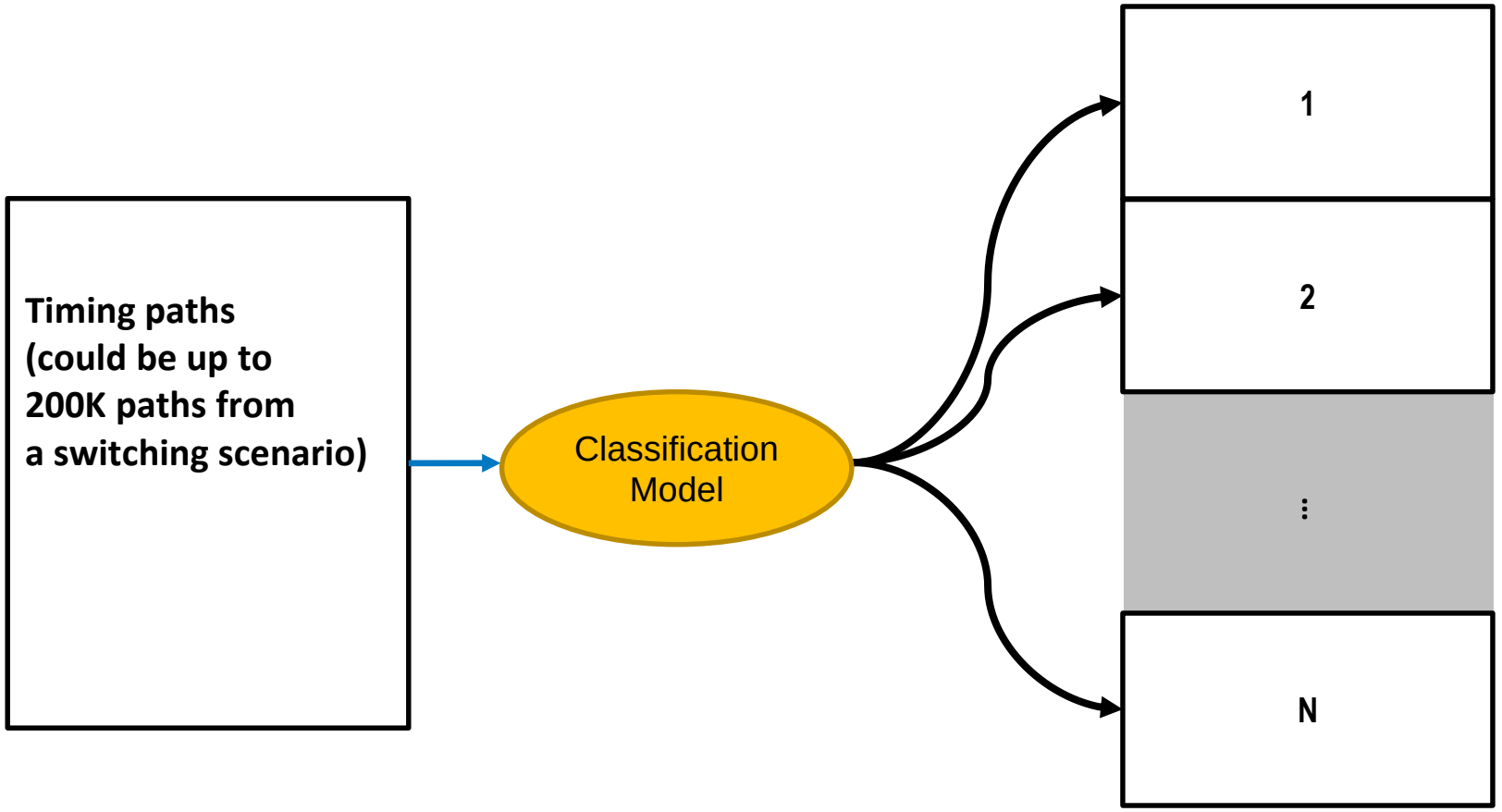
- Once the ranking order inside the top bin is achieved, weighted algorithm will be applied to determine the order of switching scenario among multiple switching scenarios.
- Each scenarios score will be calculated by the following method:
  - ***Scenarios<i>_score = Score of path<id>_from_scenario<i> * normalization term***
  - ***Score of path<id>_from_scenario<i>* defined as following:**

    split ranked path into N bins (e.g., N = 10) , the score of top-10% ranked path *= 10* , and bottom-10% ranked path *= 1*
  - ***Normalization term = 1 / log(rank_position + 1)***

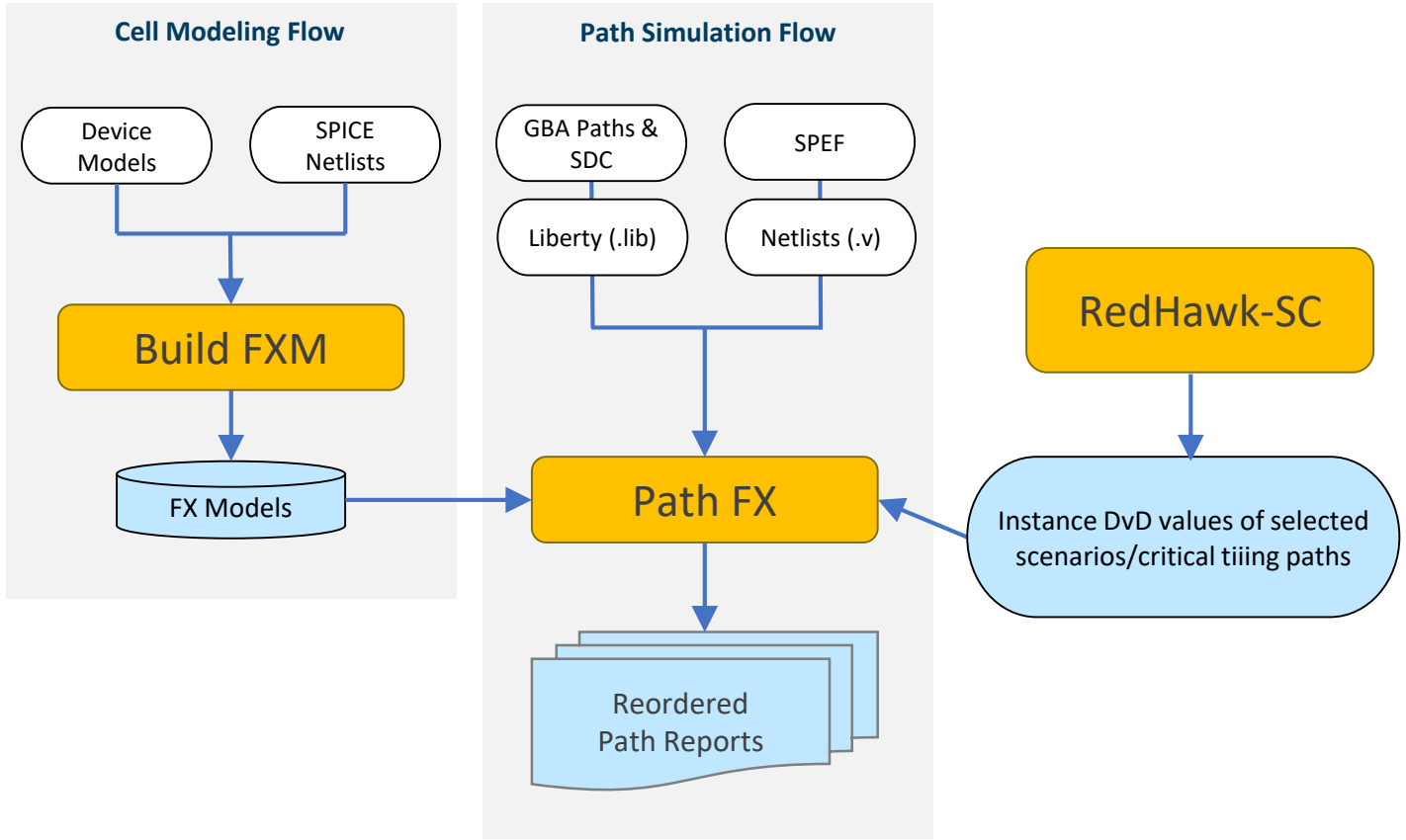


Inference Engine for Critical Path Predictor

- Bin-based classifier by classifying the timing paths to a number of bins (number of timing paths per bin could be 10K – 50K)
- From the Bin classification result, users can choose the Timing Paths in the top bin(s) with potential large delta_slack impact from DvD to run simulation with DvD waveform on Vdd/Vss pins (e.g. Path-FX) for final setup time check.



DvD aware Path Simulation Flow using Path FX



Highlights

1. Characterize FX libraries once
2. Standard inputs for timing/simulations
3. Comprehensive scenario/critical timing path coverage with ML

New Path Report

- SPICE accurate slack report
- 100x faster than MC SPICE
- Captures process and voltage variability

Model Performance of Scenario and Path Predictor for 2M Timing Paths and 20 Scenarios as Training/Validation/Test Data - Metrics

- For Bin-based Classification Step :
  - **Accuracy =** $\frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$
  - **Confusion Matrix**

Prediction

Actual		Bin 1	Bin 2
	Bin 1	90	10
	Bin 2	50	150

Bin 1 Accuracy = 90 / (90+10) = 90%
Bin 2 Accuracy = 150 / (50+150) = 75%
Avg. Accuracy = (90+150) / (90+10+50+150) = 80 %

- For Regression Step :
  - **MSE : Mean Squared Error (the smaller, the better)**
  - **R2 : R-squared (between 0~1, best:1)**
- For Bin-based Ranking :
  - **NDCG (Normalized Discounted Cumulative Gain: between 0~1, best:1) for both XGBoost and lambdaMART methods**

Model Performance of Scenario Predictor

- Top-1 bin accuracy and average accuracy on testing set of Bin-based classification model (5 bins) in **Scenario Predictor** :

Algorithms	Top-1 bin Accuracy(%)	Average Accuracy(%)	Training Time (minutes)	Testing Time (minutes)
KNN	90.1	80.2	1.1	2.2
Random Forest	92.3	86.5	16.3	0.5
XGBoost	**92.8**	**88.4**	17.5	1.8
Neural Networks	92.5	82.5	228.7 (GPU)	0.6

- MSE, R-squared and NDCG score on testing set of **Ranking model** :

Algorithms	MSE	R-squared	NDCG
XGBoost	**0.037**	**0.902**	**0.79**
Decision Tree	0.044	0.883	0.77
LambdaMART	-	-	0.77

Model Performance of Scenario (Weighted Algorithm) and Path Predictor

- Weighted algorithm on ground truth and predicted data :

Rank Position	On Ground Truth (Scenario_<index>)	On XGBoost Predicted Data	On LambdaMART Predicted Data
1	**Scenario_3**	**Scenario_3**	**Scenario_3**
2	**Scenario_15**	**Scenario_15**	Scenario_12
3	**Scenario_12**	**Scenario_12**	Scenario_15
4	**Scenario_18**	**Scenario_18**	**Scenario_18**
5	Scenario_2	Scenario_11	Scenario_13

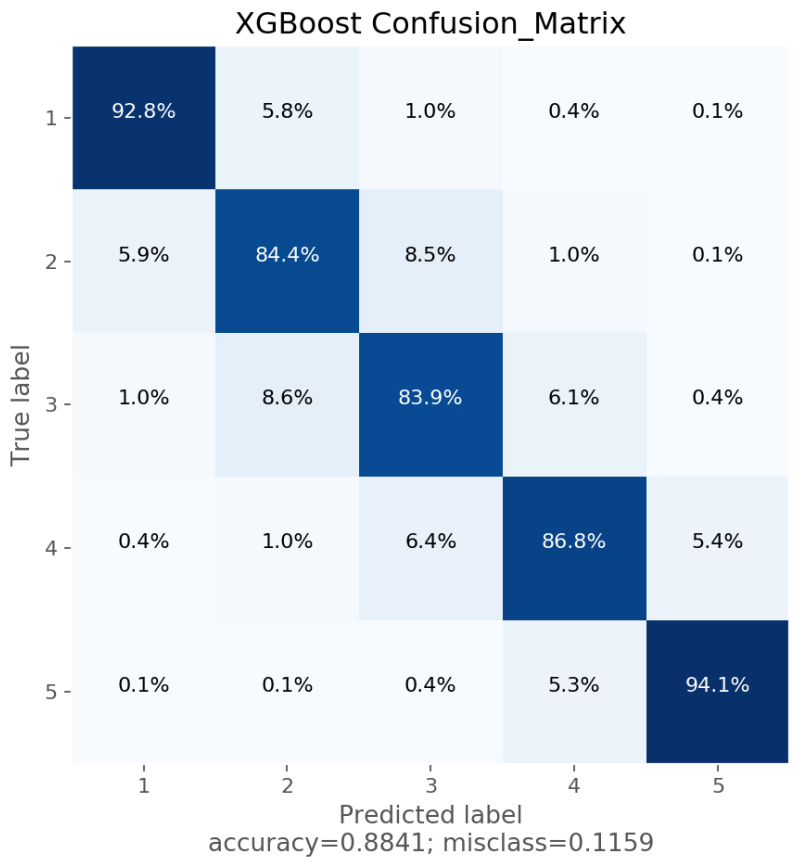
- Top-1 bin accuracy and Average Accuracy on testing set of Bin-based classification model (5 bins) in **Path Predictor** : XGBoost has the best accuracy with reasonable training/testing time

Algorithms	Top-1 bin Accuracy(%)	Average Accuracy(%)	Training Time (minutes)	Testing Time (minutes)
KNN	89.5	78.1	< 1	3.1
Random Forest	92.5	86.1	17.1	0.6
XGBoost	**93.3**	**89.3**	19.8	2.1
Neural Networks	93.2	83.8	237.2 (GPU)	0.7

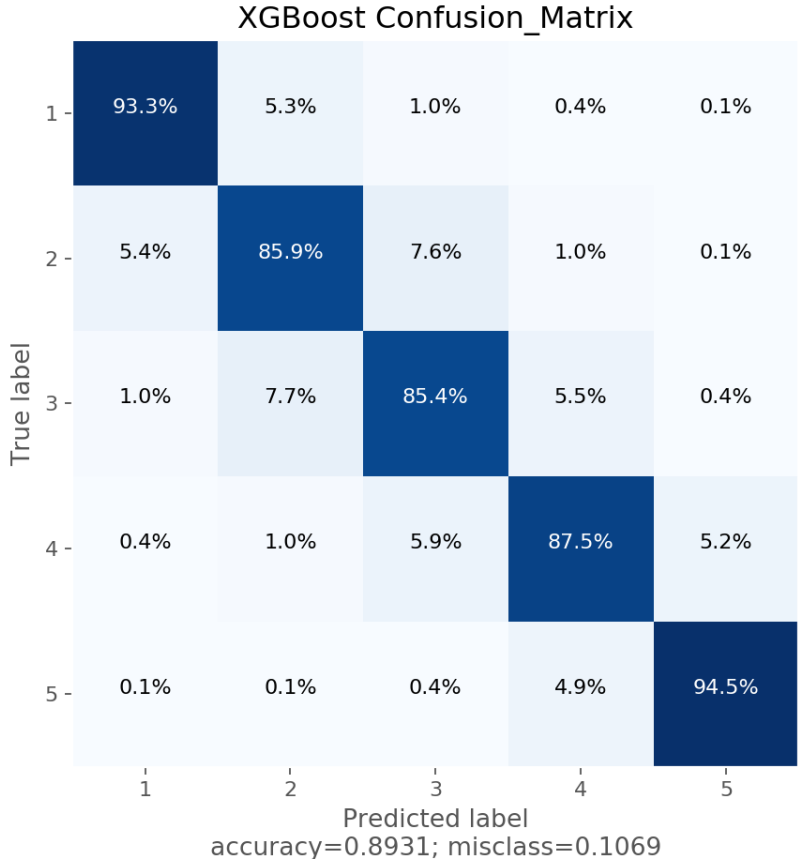
Model Performance of Scenario/Path Predictor on Accuracy Using XGBoost

- XGBoost – 5 bins of Scenario Predictor and Path Predictor

Step1 of Scenario Predictor :
Bin-based Classification Model
(17 Features)



Path Predictor :
Bin-based Classification Model
(19 Features)



Conclusions

- ML-based Timing Assistant can predict top-ranked switching scenario(s) from hundreds of possible scenarios for continuing dynamic voltage drop analysis and top-ranked timing paths of selected switching scenario(s) with DvD analysis for continuing accurate/fast timing simulation including DvD waveforms on the Vdd/Vss pins for final timing sign-off.

- The custom path-based features identified for training the inference engines (i.e. Scenario and Path Predictor) in Timing Assistant include possibly path-based effective P/G grid resistance from bumps to instance, combo peak current (Ipeak) considering rail-connected instances, path-based timing sensitivity due to voltage drop, and combined input feature from feature engineering.

- XGBoost used in both Scenario/Path Predictors have the best accuracy (> 93%) with reasonable running time.


рахмат
danke
謝謝
ngiyabonga
teşekkür ederim
Баярлалаа
спасибо
faafetai lava
thank you
gracias
tapadh leat
dank je
misaotra
matondo
paldies
grazzi
хвала
asante
manana
obrigada
kiitos
dankie
dhanyavad
hvala
maruuru
köszönöm
bedankt
dziękuję
enkosi
djere dieuf
mochchakkeram
chnorakaloutioun
gratias ago
grácies
sulpáy
go raibh maith agat
mamnun
chokrane
murakoze
tenki
obrigado
sobodi
dekuji
sukriya
kop khun krap
taiku
sagolun
arigatô
takk
dakujem
trugarez
grazie
tanemirt
rahmet
terima kasih
dhanyavadagalu
shukriya
мерси
najis tuke
diolch
merci
감사합니다
xièxie
ευχαριστώ
谢谢