UHS

BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

Induction of Labor from Decision to Failure: Feature Selection and Predictive Modeling Approach

Anemone Kasasbeh, PhD
Lead Data Scientist
United Health Services Hospitals Inc.,

Anemone.kasasbeh@nyuhs.org
@anemonekasasbeh

November 15th, 2023

Introduction

- Induction of Labor (IOL): Widely used artificial interventions to end expectant management
- IOL is one of the recommended practices for terminating a pregnancy
- Several guidelines available (WHO, NICE, Queensland Health, ACOG)
- Suffer from vagueness, generality, inconsistency, and prioritization.
- This can lead to early induction of labor for many women whom their pregnancy would have been better with expectant management

Problem Description

- Failure in IOL results in a C-section
    - Contribute to a higher rate of cesarean deliveries
- One-fifth of IOLs fail to result in vaginal birth
- One C-section, high Probability for Future C-sections

Objective

- A need for a robust prediction model that can help medical professionals with making the right decision at the right stage
- There is no predictive model.
- No Current Calculators and Some Proposed are Limited

Model Goal/Objective/Input/Output

- Goal: Improve Maternal and Neonatal Healthcare Outcomes
- Objective: Develop an Integrated Framework to Predict Patients' Induction of Labor Failure or Success
- Inputs: 779 Features
- Output: Vaginal/C-section

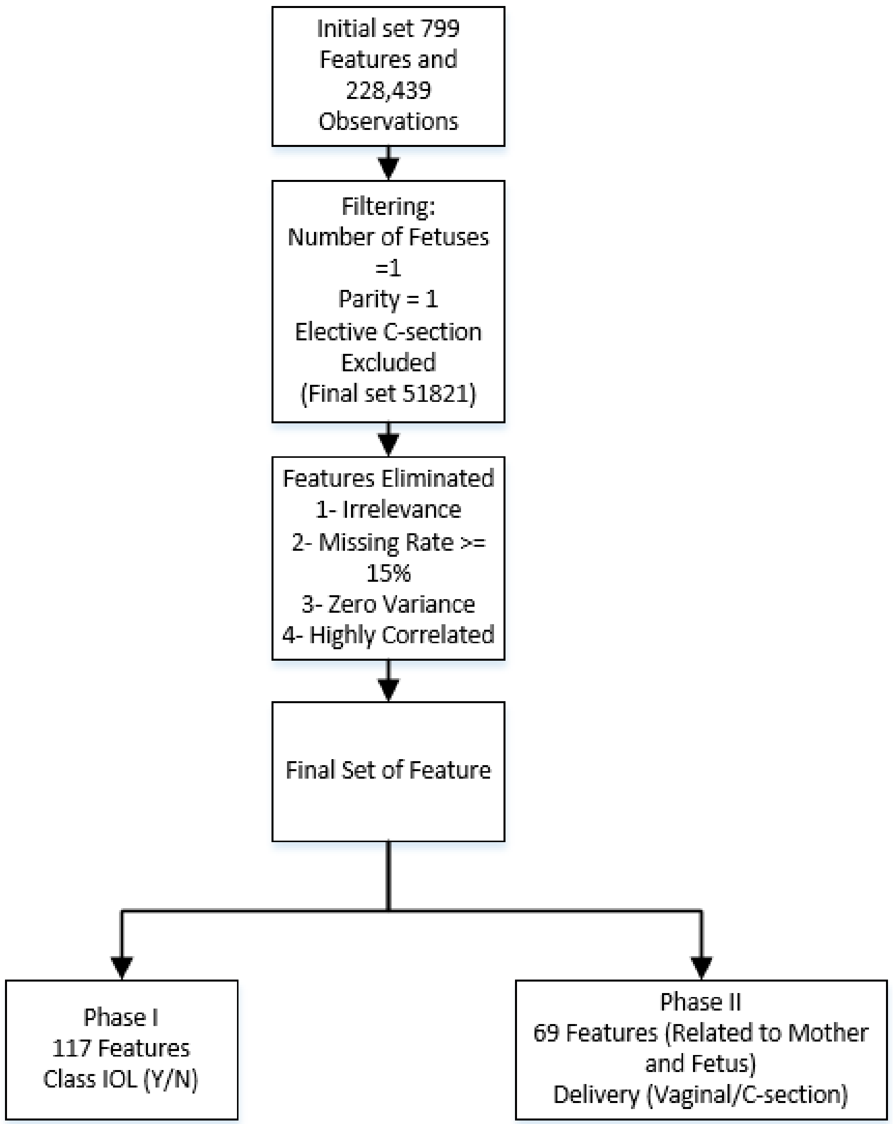
Indicators to Artificially Induce Labor based on Guidelines and Literature

Variables	Anticipated Effect
Fetal Position	It depends on the position of the baby. They usually do not recommend Breech Position Pregnancy to be induced. Whereas, vertex position has higher odds of successful induction of labor.
Bishop Score	Lower Bishop score lead to failure of induction.
Gestational Age	Post term or full term both increase the odds of successful induction of labor.
Cervical Ripening	Unfavorable ripened cervix can lead to a C-section
Diabetes - gestational and type I and II	If women were induced only for gestational diabetes, odds are high for a failure of the induction
Fetal macrosomia	Fetal Macrosomia should not be a factor to induce labor
History of precipitate labor	Variations in literature
Hypertension, pre-eclampsia	If only reason failure of IOL is anticipates
Maternal ethnicity	Unknown
Maternal cardiac disease	Could lead to a C-section since IOL take long hours
Maternal request	Maternal request with no other reasons most likely will end up in a cesarean
Epidural Administration	Some literature mention that Epidural could lessen the strength of the push/contraction which could lead to a Cesarean
Prolonged pregnancy	Anticipated to end in a vaginal birth. However, gestational age is an estimated value.
Premature Rupture of Membranes - at term	Women with PROM at term should be offered IOL, however, in terms of results, it is still unknown. It is also worth mentioning, that there is an inconsistency in the guideline on when is term exactly (37+00 weeks, NICE 2008) whereas (41+00 weeks, WHO 2011)
Premature Rupture of Membranes - preterm	There is a variation between guidelines.
Twin pregnancy - Uncomplicated	If only reason failure of IOL is anticipated

Data Description

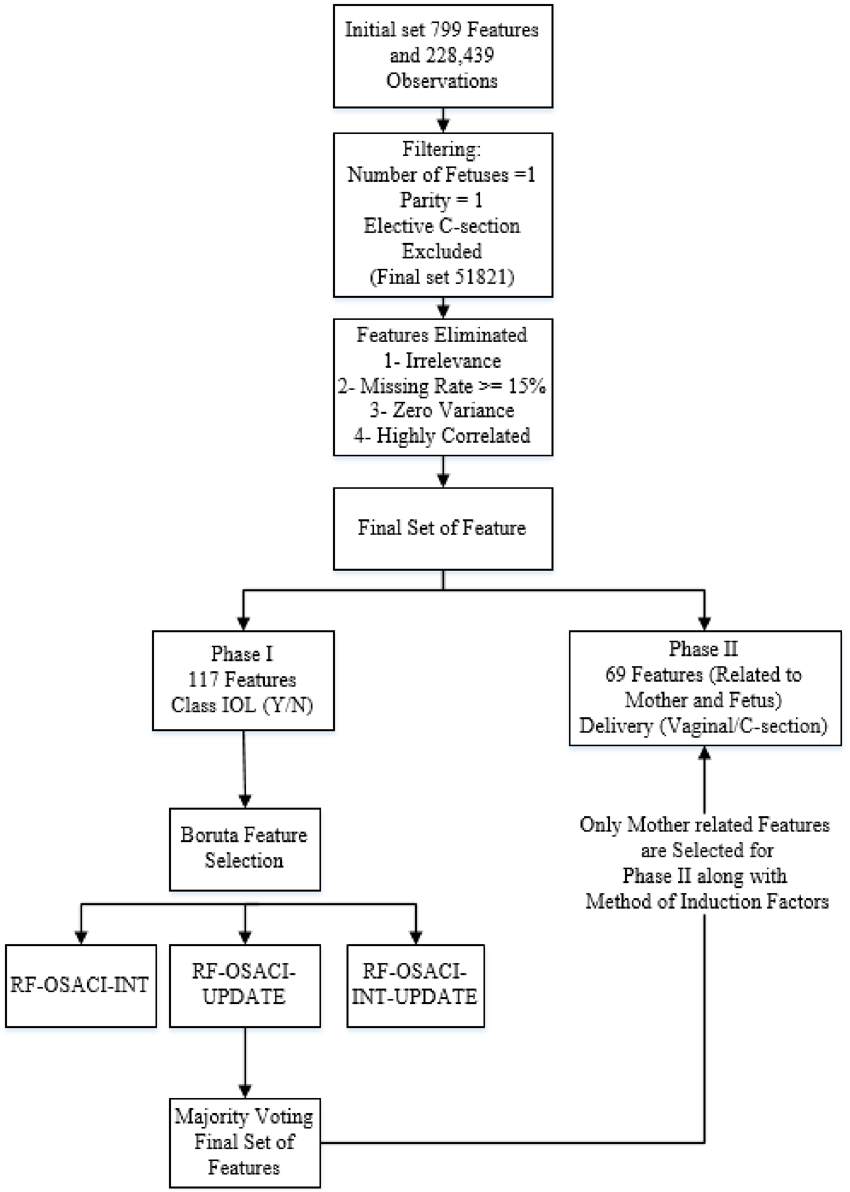
- National Institute of Child Health and Human Development (NICHD) through Data Specimen Hub
- Original Data:12 hospitals, 779 factors, and 228,439 pregnancies
- Medical History, Perinatal, Maternal, Delivery, Neonatal…etc

Study Framework : Features Elimination and Features Selection Outline



Phase I: IOL from Pregnancy to Decision

- Feature Elimination
- Feature Selection using Stacking and Metaheuristics



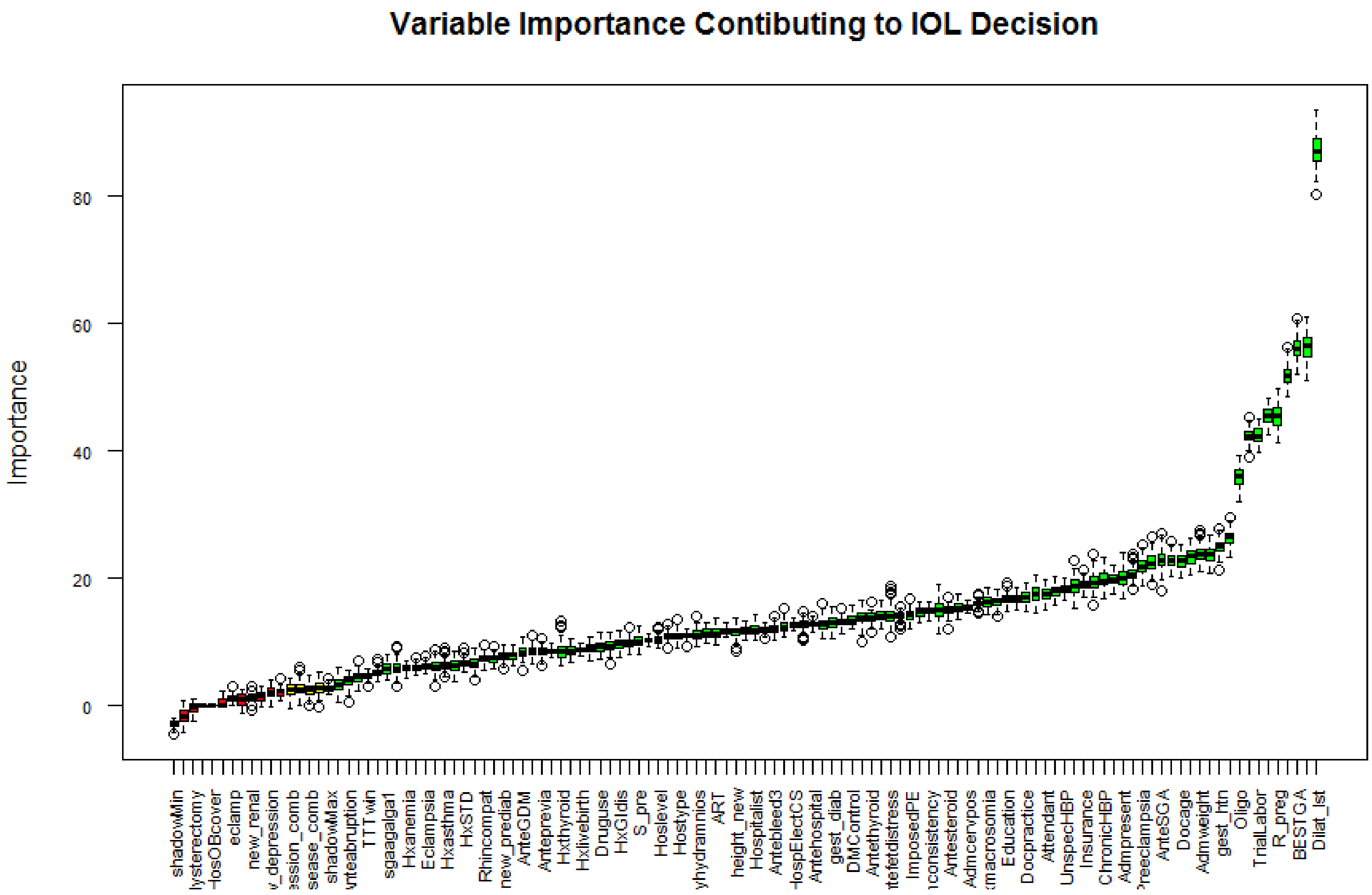
Sensitivity Scores and Dimensionality Reduction of the Hybrid Feature Selection Model

Method	Accuracy	Sensitivity	Specificity	CPU Time	Dimensionality Reduction
RF-Boruta	81.8%	88.1%	71%	17 mins	11%
RF-OSACI-Init	80%	69%	87%	16 mins	60%
RF-OSCAI-Update	79%	70%	85%	18 mins	60%
RF-OSCAI-Init_Update	80.5%	70%	87%	20 mins	62%

Final set of Predictors Selected by Boruta (Top 30 out of 101)

Number	Variable	Definition	Mean Importance
1	Dilat_lst	Repeated measures: Dilation of first exam	**87.3**
2	Admcontract	Admission to L&D: Number of contractions per 10 minutes	**56.2**
3	BESTGA	Admission to L&D: Best estimate gestational age (weeks)	**56.2**
4	ROMmeth	Labor and delivery summary: Method of ROM	**51.9**
5	ROM	Derived variable from chart: PROM, exclusive of spontaneous labor	**45.5**
6	R_preg	Number of repeat pregnancies	**45.4**
7	TrialLabor	Trial of labor (defined as vaginal deliveries or c-section with 2 records in repeated measures database)	**42.5**
8	Numvagex	Repeated measures: Number of exams/patient	**42.4**
9	Oligo	Prenatal history: Oligohydramnios	**35.9**
10	Docspecialty	Physician characteristics: Specialty	**26.4**
11	gest_htn	Derived variable: Gestational hypertension	**24.9**
12	Admweight	Admission to L&D: Weight (kg) at admission	**23.8**
13	PROM_new	Derived variable: PROM	**23.7**
14	Birthweight1	Newborn information: Birth weight (g) (1)	**23.5**
15	momrace_new	Maternal race, some missing data replaced with data from repeat pregnancies	**22.9**
16	AnteSGA	Prenatal history: Intrauterine growth restriction	**22.9**
17	Docage	Physician characteristics: Age	**22.8**
18	Delfetalpos	Labor and delivery summary: Fetal position	**22.5**
19	Preeclampsia	Prenatal history: Preeclampsia/HELLP	**21.9**
20	Momage	Maternal demographic: Maternal age	**20.8**
21	Admpresent	Admission to L&D: Presentation	**20.1**
22	GestHBP	Prenatal history: Prenatal gestational hypertension	19.8
23	ChronicHBP	Medical history: Chronic hypertension	19.8
24	Presentdel	Labor and delivery summary: Fetal presentation at delivery	19.4
25	Insurance	Maternal demographic: Insurance type	19.1
26	AnteLGA	Prenatal history: Large for gestational age	18.8
27	UnspecHBP	Prenatal history: Unspecified hypertension	18.3
28	Hosinsurance	Annual insurance premium	18.0
29	Attendant	Labor and delivery summary: Who delivered?	17.6
30	PROM	Prenatal history: Premature rupture	17.5

Boruta Select Features Box Plot

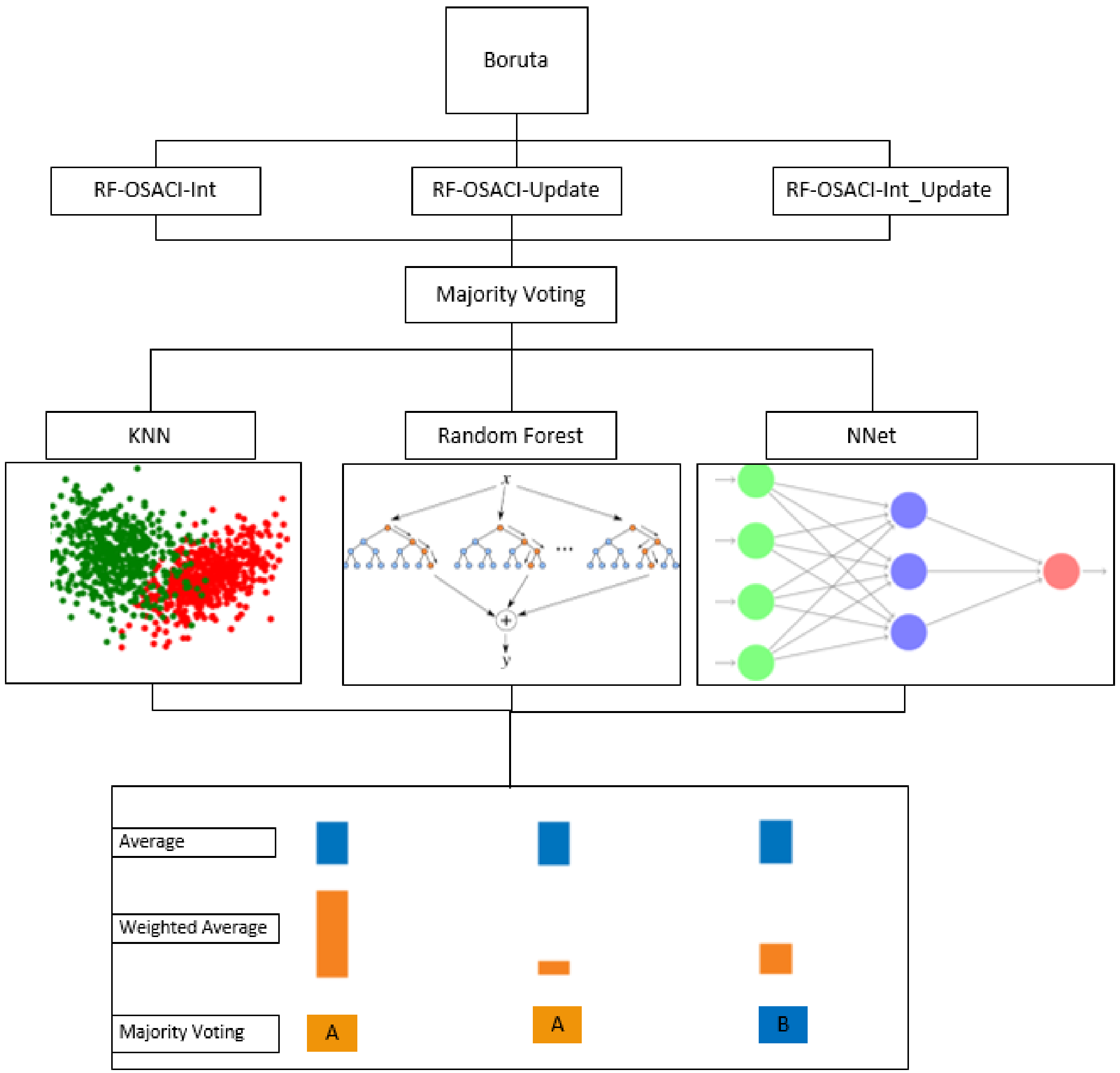


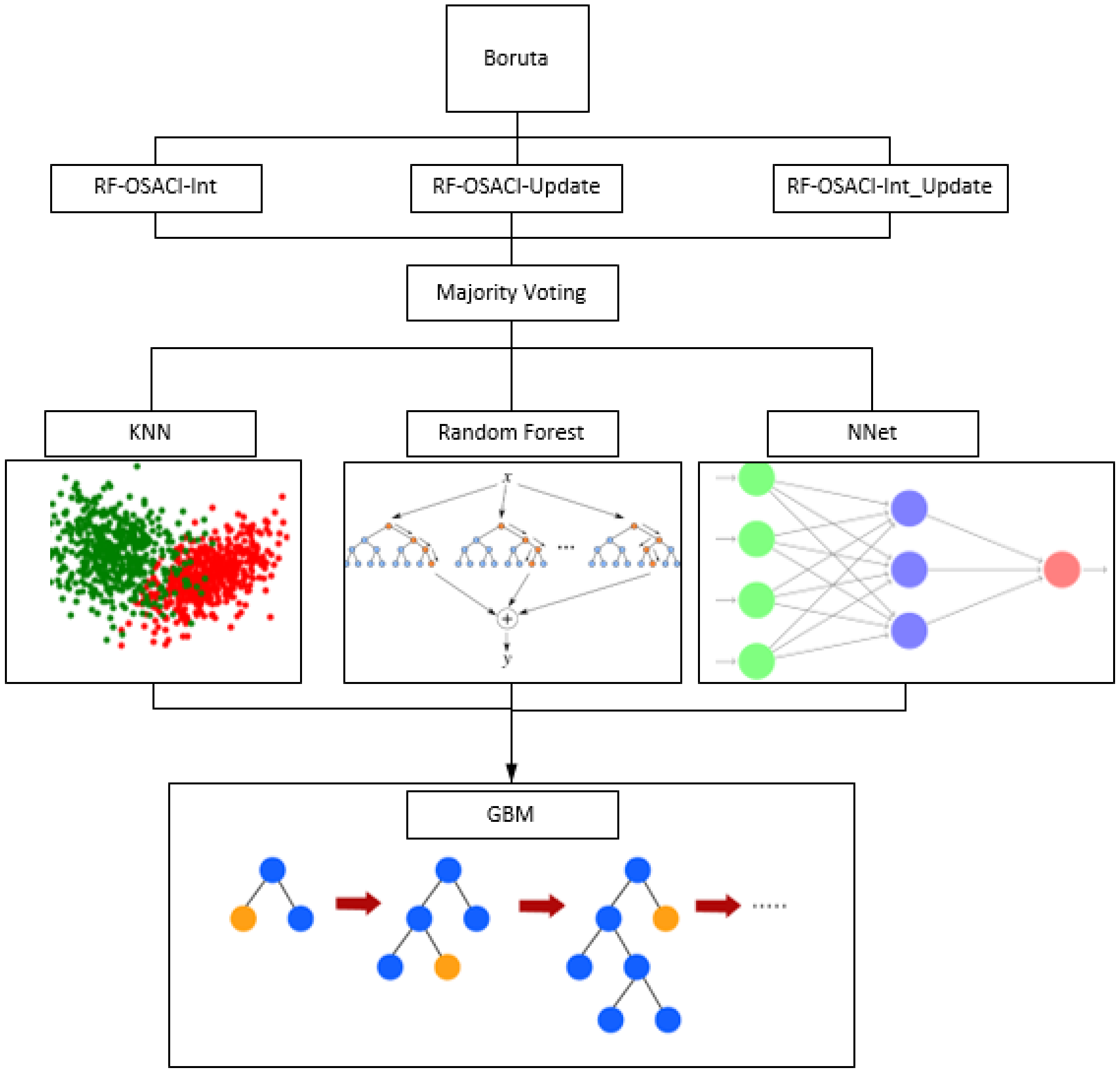
Final set of Predictors Selected by RF-OSACI Three Variants (Top 31 out of 46)

Number	Final	Definition
1	BESTGA	Best estimate gestational age (weeks)
2	Admcontract	Number of contractions per 10 minutes
3	Dilat_lst	Dilation of first exam
5	**Docage**	**Physician characteristics: Age**
6	**Hosinsurance**	**Annual insurance premium**
7	**Hosnumdel**	**Number of deliveries in 2006**
8	Marital	Marital status
9	ChronicHBP	Medical history: Chronic hypertension
10	Hxheartdis	Medical history: Heart disease
11	Hxrenaldis	Medical history: Renal disease
12	HxGIdis	Medical history: Gastrointestinal disorder
13	Hxseizure	Medical history: Seizure
14	HIV	Medical history: HIV
15	ECV	Prenatal history: External cephalic version

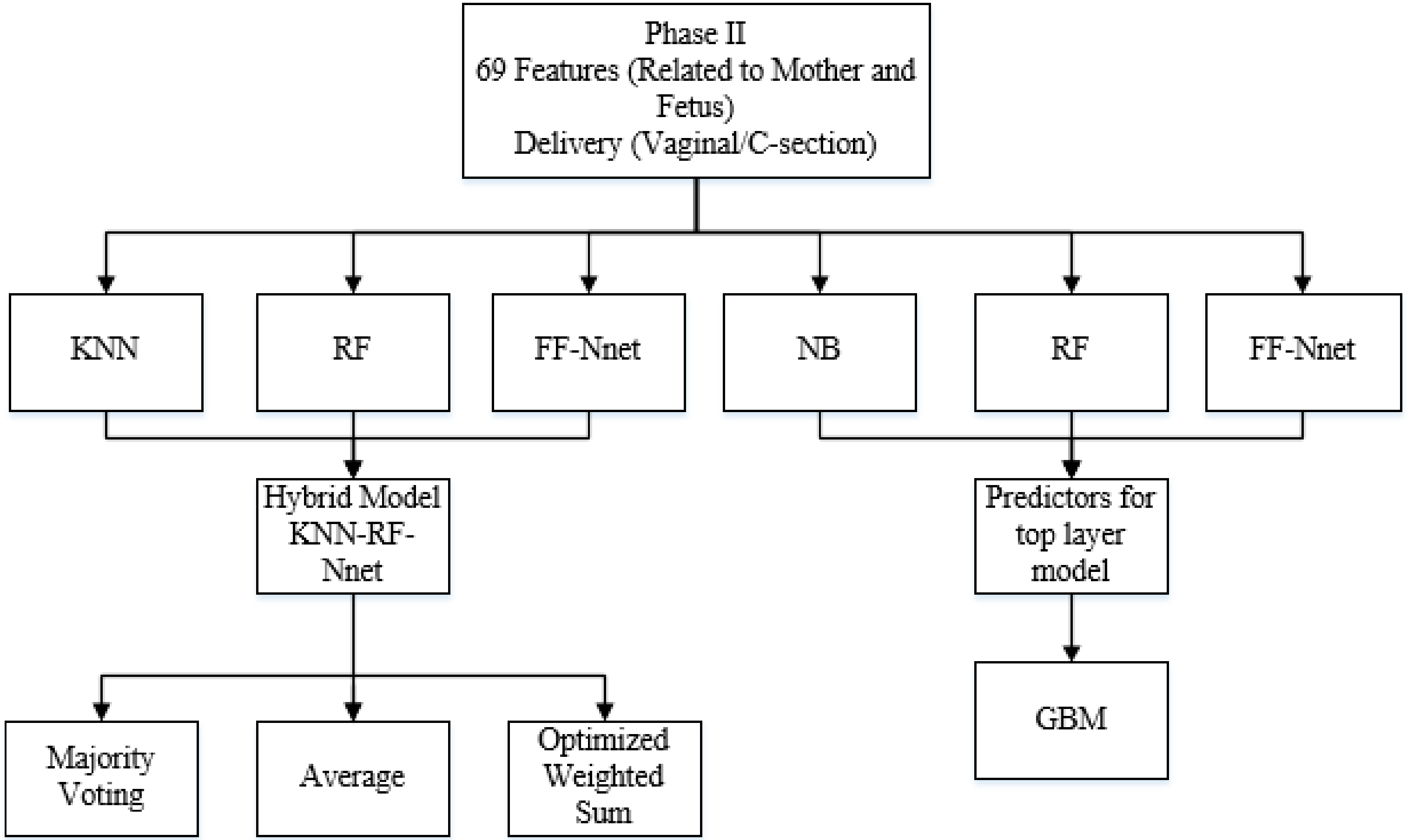
Number	Final	Definition
16	Antehospital	Prenatal history: Antenatal hospital admission
17	Antebleed3	Prenatal history: Bleeding in the third trimester
18	Antethyroid	Prenatal history: Prenatal thyroid disease
19	AnteGDM	Prenatal history: Gestational diabetes
20	DMControl	Prenatal history: Most serious diabetes control
21	Anteabruption	Prenatal history: Abruptio placentae
22	ThreatenedPB	Prenatal history: Threatened preterm birth
23	PROM	Prenatal history: Premature rupture
24	Rhincompat	Prenatal history: Rh incompatibility
25	CPD	Prenatal history: Cephalopelvic disproportion (CPD)
26	TTTwin	Prenatal history: Twin-to-twin transfusion
27	ROMmeth	Labor and delivery summary: Method of ROM
28	Delfetalpos	Labor and delivery summary: Fetal position
30	Hostype	Type of hospital
31	HospElectInd	Elective induction prior to 41 weeks

Proposed Models in Phase II





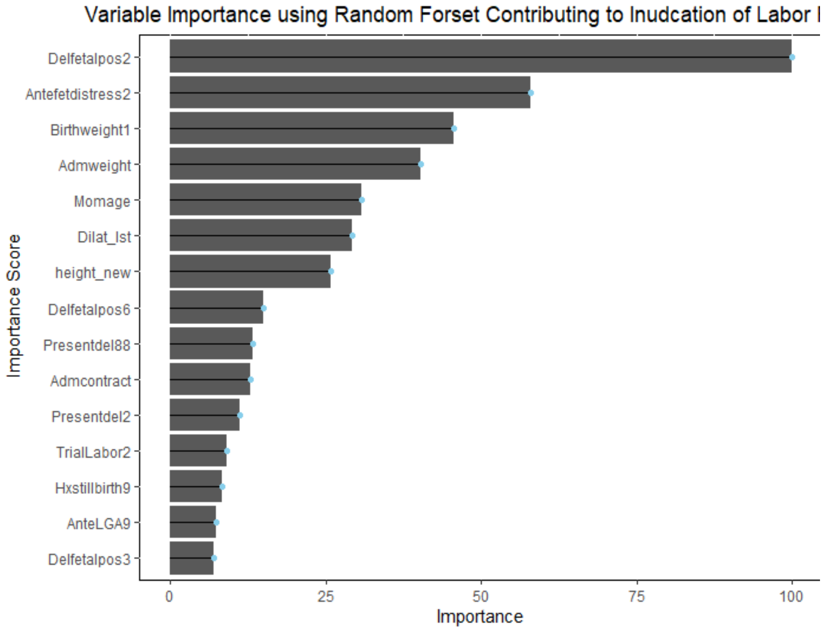
Phase II: IOL from Pregnancy to Failure

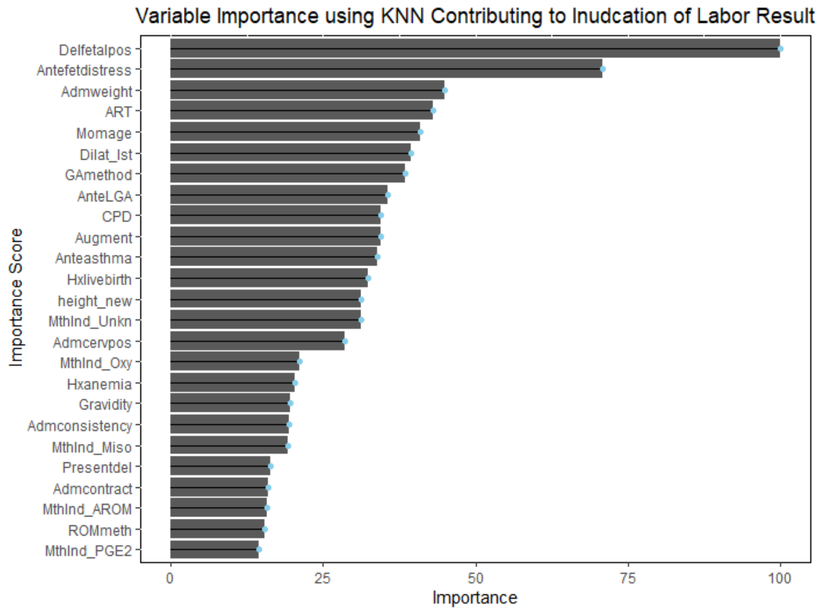


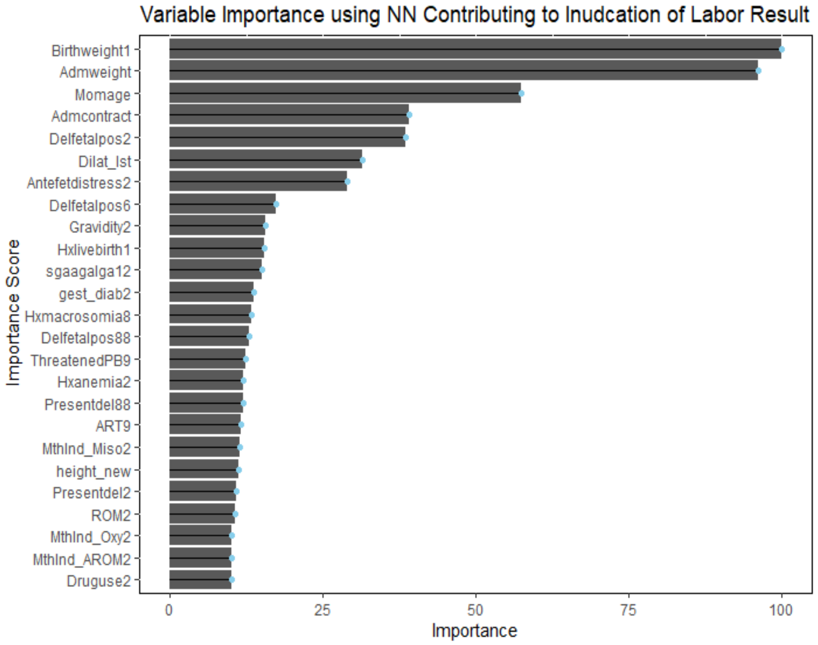
Performance and Computational Cost Metrics

Model	Accuracy	Sensitivity	Specificity	Normalized CPU Run Time (min)
RF	94.1%	84.7%	94.6%	36
KNN	75%	88%	61%	19
FF-Nnet	83.81%	83.52%	84.09%	5
Average Hybrid Model	93.78%	98.25%	89.27%	37
Optimized Weighted Sum Hybrid Model	94.75%	90.86%	93.62%	39
Majority Voting Hybrid Model	93.66%	97.53%	89.75%	37
Hybrid Model (RF-NB-Nnet, GBM Top Layer)	**95.5%**	**93.6%**	**90.4%**	**36**

Top Important Features Selected







Conclusion and Contribution

- First Research to
  - Study Historical Driving Factors behind IOL
  - Use High Dimensional Data in this Domain
  - Use Metaheuristics for Feature Selection in a Multi-step Hybrid feature Selection Model
  - Apply Machine Learning Algorithms other than LR and RF
  - Develop Hybrid Models in this Domain

Contribution

- Previous Studies pre-selected the variables
  - Our study used advanced and regress approaches to select the most contributing factors.
- Proven that Non-clinical Drivers are Contributing to the IOL decision
- Proven that variations and vagueness in guidelines in the IOL guidelines can Terminate Expectant Management Early
- A large number of contributing factors proven the urgent need for a user-friendly predictive model to help assess the decision of IOL
- Developed and Proposed Models to Improve maternal and fetus health by streamlining the IOL decision using demonstrated predictive models and identifying patients who might benefit from the IOL decision.

Future Directions

- Retrain the Models for a Specific Hospital Data (UHS)
  - Develop AI algorithms that are trained on diverse and custom datasets and specifically designed to address the unique needs and characteristics of marginalized patient populations.
- Embed the Final Model in Production with a User Friendly Interface
- Expand the Study to Address Post Pregnancy Complications after IOL

Publications

- Induction of Labor from Decision to Delivery Using Hybrid Predictive Models based on a Metaheuristic Feature Selection Approach

Anemone Kasasbeh1, Liliane El-Kassis, and Hiroki Sayama (Ready to Submit)

International Journal of Clinical Practice

- Web-based Healthcare Delivery Integrated System to Forecast COVID-19 Hospitalizations in a Marginalized Patient Population: A Case Study in Broome County, New York,

A Kasasbeh, M Yildirim, A Booth, N Khan, H Sayama (Submitted)

Journal of Environmental and Public Health

- Modelling the Impact of Transportation Availability and Travel Distance on Healthcare Outcomes: A Bagged Random Forest Approach

A Kasasbeh, M Yildirim, A Booth, N Khan, H Sayama

IISE Annual Conference Proceedings (2023)

- Influential Factors for Failure to Show up for a Postpartum Visit

A Kasasbeh, M Yildirim, A Booth, MT Khasawneh

IISE Annual Conference & Expo 2019, 883-889 (2019)

- Crash severity prediction using a series of artificial neural networks

A Kasasbeh, R Shabbar, D Santos

IISE Annual Conference. Proceedings, 443-448 (2018)

- Charging station allocation for electric vehicle network using stochastic modeling and grey wolf optimization

R Shabbar, A Kasasbeh, MM Ahmed

Sustainability 13 (6), 3314 (2021)

- Crash Analysis Using Artificial Neural Network and Decision Tree

A Kasasbeh, R Shabbar

Industrial and Systems Engineering Review (ISER) (2017)

- Demand forecasting for inventory control: A case study on automotive spare parts in Saudi Arabia

N Khan, A Kasasbeh, R Alkhasawneh

- IISE Annual Conference & Expo 2018 (2018)
- Proactive Event Management using ANN with PSO Prediction in Transport Processes

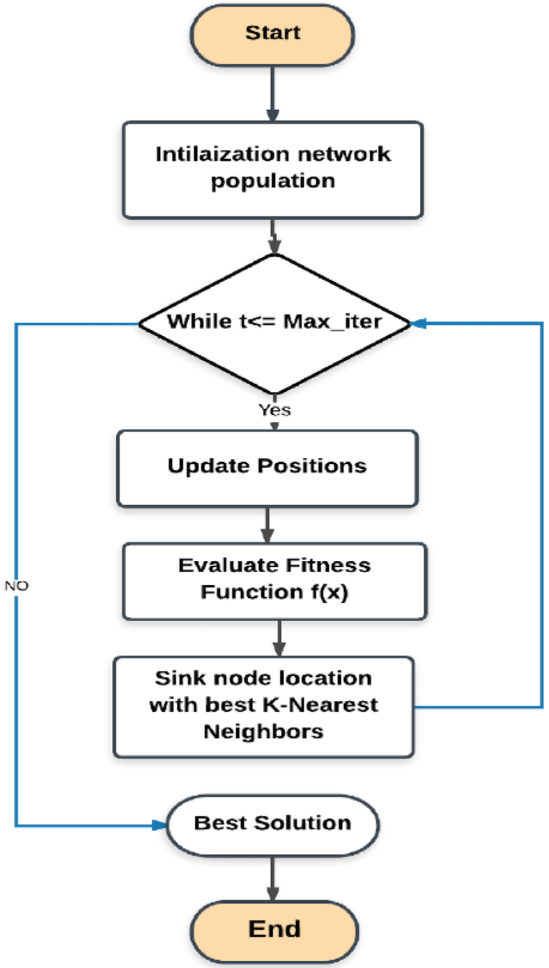
R Shabbar, A Kasasbeh (2017)

THANK YOU!

Questions?

Anemone.kasasbeh@nyuhs.org
@anemonekasasbeh

KNN



For KNN, the tuning parameter is k, the number of nearest neighbors to consider. For example, tuneGrid = data.frame(k = 1:10) specifies a range of values for k.
metric: the performance metric to use for model selection. For classification, common metrics include accuracy (metric = "Accuracy") and area under the receiver operating characteristic curve (metric = "ROC").
trControl: a control object that specifies the cross-validation method and other options for model tuning. For example, trControl = trainControl(method = "cv", number = 10) specifies 10-fold cross-validation for model tuning.

ARIMA

- ARIMA.Param   <- c(7, 1, 2)
- Include.Mean   <- FALSE
- Include.Constant <- FALSE
- include.Drift   <- TRUE

- # Fit ARIMA model
- TimeSeries.Fit <- Arima(TimeSeries.Vol,
-                         order   = ARIMA.Param[1:3],
-                         include.mean   = Include.Mean,
-                         include.constant = Include.Constant,
-                         include.drift   = include.Drift,
-                         xreg   = NULL,   # Exogenous variable (should be a vector not a data frame)
-                         method   = "ML")

ARIMA

order
A specification of the non-seasonal part of the ARIMA model: the three integer components (p, d, q) are the AR order, the degree of differencing, and the MA order.

seasonal
FALSE

include.mean
TRUE

The argument transform.pars will be set to FALSE if any AR parameters are fixed. A warning will be given if transform.pars is set to (or left at its default) TRUE. It may be wise to set transform.pars = FALSE even when fixing MA parameters, especially at values that cause the model to be nearly non-invertible.

init
Not set. Missing values will be filled in, by zeroes except for regression coefficients. Values already specified in fixed will be ignored.

method
fitting method: maximum likelihood

RF

BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

- ntree: the number of decision trees to build 500.
- mtry: the number of variables to select at each split. The default value is the square root of the number of predictor variables.
- replace: a logical value indicating whether sampling of the data is done with or without replacement. TRUE.
- sampsize: the number of samples to draw from the data for each tree. nodesize: the minimum size of terminal nodes. Set to1.
- importance: a logical value indicating whether to calculate and return variable importance measures. FALSE.
- proximity: a logical value indicating whether to calculate and return a proximity matrix. FALSE.
- do.trace: a logical value indicating whether to print progress messages during the model fitting process. FALSE.
- keep.inbag: a logical value indicating whether to keep the "in-bag" samples for each tree. FALSE.
- classwt: a vector of weights for each class to address class imbalance. NULL.
- subset: a logical vector specifying a subset of observations to be used for the model training. NULL.

RF vs Bagged RF

- The main difference is in the way they create the ensemble of decision trees.
- In RF, each tree is trained on a randomly selected subset of features, which helps to reduce the correlation between the trees and improve the overall accuracy of the model. This process is known as "feature bagging". Additionally, in RF, each tree is grown to its maximum depth without any pruning, which can lead to overfitting.
- In Bagged RF, on the other hand, each tree is trained on a bootstrap sample of the original data set. This means that some of the original data points may not be included in each bootstrap sample, and some may be included multiple times.
- This process is known as "bootstrap aggregating" or "bagging". Bagging helps to reduce the variance of the model and prevent overfitting. Additionally, in Bagged RF, each tree is grown with pruning, which helps to further reduce overfitting.
- Another difference is in the number of features that are randomly selected for each tree. In RF, the number of features to be considered at each split point is usually set to the square root of the total number of features. In Bagged RF, all features are considered for each split point.
- In summary, while both RF and Bagged RF use decision trees as base learners and are ensemble learning algorithms, the main differences between the two are the use of feature bagging in RF and bootstrap aggregating in Bagged RF, the depth of the trees and the feature selection process. In general, RF tends to perform better than Bagged RF on most datasets, but Bagged RF can be more robust to noise in the data.

Bagging

- Bagging (Bootstrap Aggregating) is a popular ensemble learning method that combines multiple base models trained on different subsets of the original training data. The key parameters in Bagging are:

- base_estimator: This parameter specifies the base estimator or model to be used for bagging. It is typically set to a decision tree, but other models such as SVM, KNN, or neural networks can also be used.

- n_estimators: This parameter specifies the number of base models or estimators to be trained. A larger number of estimators can improve performance but also increases computation time.

- max_samples: This parameter specifies the maximum number of samples to be used for training each base estimator. It can be set to a fraction or a percentage of the total number of training samples.

- max_features: This parameter specifies the maximum number of features to be used for training each base estimator. It can be set to a fraction or a percentage of the total number of features.

- bootstrap: This parameter specifies whether or not to use bootstrap sampling to randomly select the training samples for each base estimator. By default, it is set to True, which means that bootstrap sampling is used.

- bootstrap_features: This parameter specifies whether or not to use bootstrap sampling to randomly select the features for each base estimator. By default, it is set to False, which means that all features are used for each base estimator.

Stacking

- Average we can do averaging on the probabilities of observations to be in wither of these binary classes.
- Majority Voting: In majority voting, we’ll assign the prediction for the observation as predicted by the majority of models. Since we have three models for a binary classification task, a tie is not possible.
- Weighted Average: Instead of taking simple average, we can take weighted average. Generally, the weights of predictions are high for more accurate models. Let’s assign 0.5 to logistic regression and 0.25 to KNN and random forest each.

Boruta

- pValue

confidence level. Default value s

- mcAdj

if set to TRUE, a multiple comparisons adjustment using the Bonferroni method will be applied. Default value

- maxRuns

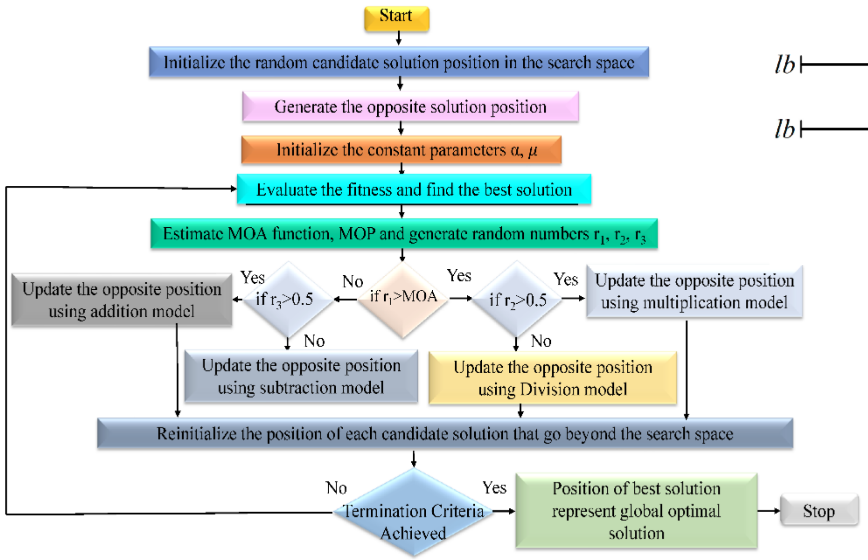
500

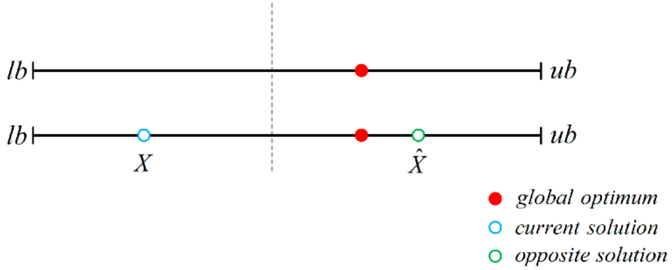
- doTrace
- verbosity level. 0 means no tracing, 1 means reporting decision about each attribute as soon as it is justified, 2 means the same as 1, plus reporting each importance source run, 3 means the same as 2, plus reporting of hits assigned to yet undecided attributes.

- holdHistory
- if set to TRUE, the full history of importance is stored and returned as the ImpHistory element of the result. Can be used to decrease a memory footprint of Boruta in case this side data is not used, especially when the number of attributes is huge; yet it disables plotting of such made Boruta objects and the use of the TentativeRoughFix function.

- getImp
- function used to obtain attribute importance. The default is getImpRfZ, which runs random forest from the ranger package and gathers Z-scores of mean decrease accuracy measure. It should return a numeric vector of a size identical to the number of columns of its first argument, containing importance measure of respective attributes. Any order-preserving transformation of this measure will yield the same result. It is assumed that more important attributes get higher importance. +-Inf are accepted, NaNs and NAs are treated as 0s, with a warning.

OSACI





Algorithm 1: OBL Initialization

Input:
S: No. of candidates;
d: No. of original features;

Procedure:
Generate a set of $S/2$ random initial candidates, $\mathbf{Z}_r \leftarrow \{\mathbf{z}_r^1, \mathbf{z}_r^2, \dots, \mathbf{z}_r^{S/2}\}$;
Allocate memory for a set of $S/2$ opposite candidates, $\tilde{\mathbf{Z}} \leftarrow \{\tilde{\mathbf{z}}^1, \tilde{\mathbf{z}}^2, \dots, \tilde{\mathbf{z}}^{S/2}\}$;
for $s = 1$ to $S/2$ **do**
  for $j = 1$ to d **do** // *Generate opposite candidate* $\tilde{\mathbf{z}}^s$
    $\tilde{z}_j^s \leftarrow 1 - z_{r,j}^s$;
  end for
end for
$\mathbf{Z} \leftarrow \{\mathbf{Z}_r \cup \tilde{\mathbf{Z}}\}$; // *Initial candidates*

Output: Initial candidates, $\mathbf{Z}$.

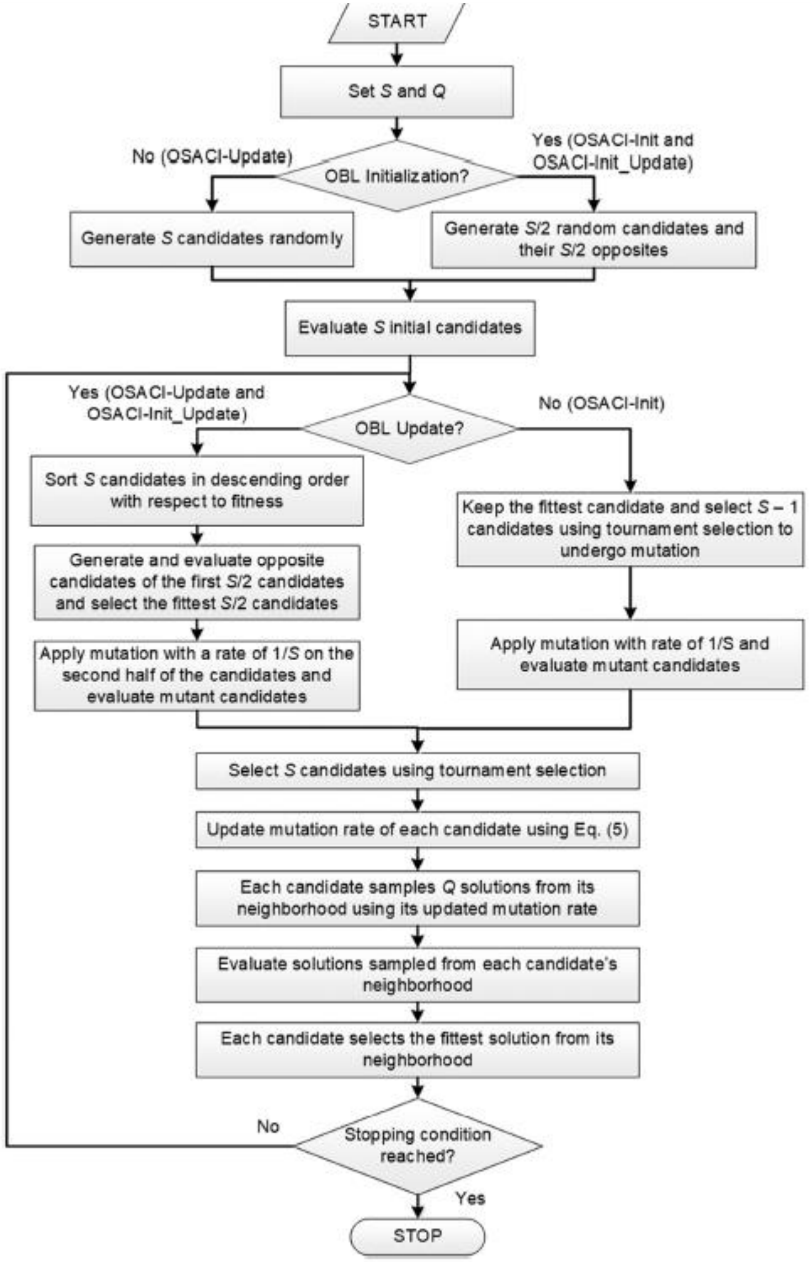
The stopping condition in SACI is met only when there is no significant improvement in the minimum and maximum fitness values of the cohort in addition to the best fitness value attained for a specific number of successive learning attempts, τ_{max}.

Algorithm 2: OBL Update

Input:
$\mathbf{Z}$: Current candidates;
S: No. of candidates;
d: No. of original features;

Procedure:
Sort candidates $\{\mathbf{z}^1, \mathbf{z}^2, \dots, \mathbf{z}^S\}$ in descending order with respect to fitness values;
Allocate memory for $S/2$ opposite candidates, $\tilde{\mathbf{Z}} \leftarrow \{\tilde{\mathbf{z}}^1, \tilde{\mathbf{z}}^2, \dots, \tilde{\mathbf{z}}^{S/2}\}$;
Allocate memory for $S/2$ mutant candidates, $\mathbf{Z}_m \leftarrow \{\mathbf{z}_m^{(S/2)+1}, \mathbf{z}_m^{(S/2)+2}, \dots, \mathbf{z}_m^S\}$;
for $s = 1$ to $S/2$ **do**
  for $j = 1$ to d **do** // *Generate opposite candidate* $\tilde{\mathbf{z}}^s$
    $\tilde{z}_j^s \leftarrow 1 - z_j^s$;
  end for
end for
Evaluate opposite candidates $\tilde{\mathbf{Z}}$;
Select the fittest $S/2$ candidates, $\tilde{\mathbf{Z}}_{best}$, from $\{\mathbf{z}^1, \mathbf{z}^2, \dots, \mathbf{z}^{S/2}, \tilde{\mathbf{z}}^1, \tilde{\mathbf{z}}^2, \dots, \tilde{\mathbf{z}}^{S/2}\}$;
for $s = (S/2)+1$ to S **do** // *Generate S/2 mutant candidates*
  Create mutant candidate, $\mathbf{z}_m^s$, by applying mutation with rate of $1/S$ on $\mathbf{z}^s$;
end for
$\mathbf{Z} \leftarrow \{\tilde{\mathbf{Z}}_{best} \cup \mathbf{Z}_m\}$; // *New candidates*

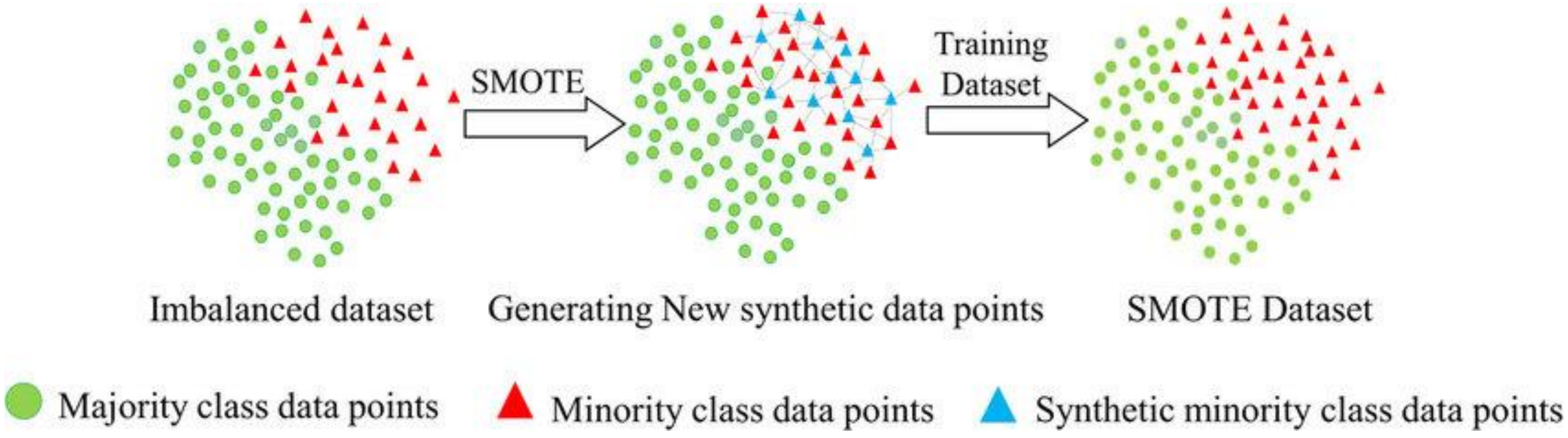
Output: New candidates, $\mathbf{Z}$, in current learning attempt.




```
OSACI(OSACIVariant = "OSACI-Init",        # "OSACI-Init", "OSACI-Update", or "OSACI-Init_Update"
      L = 10,                             # Max. no. of learning attempts
      S = 8,                              # No. of candidates
      d = ncol(TrainFeatures),            # No. of features
      Q = 10,                             # No. of quality variations (solutions sampled from neighborhood)
      k = 2,                              # Tournament size (Default = 2)
      EPSILON = 0,                        # Convergence tolerence (Default = 0)
      TauMax = Inf,                       # No. of successive learning attempts for cohort saturation (Default = Inf)
      lambda = 0.9                        # Trade-off factor in fitness function (Default = 0.9)
)
```

SMOTE

- SMOTE (Synthetic Minority Over-sampling Technique) is a popular oversampling method used in machine learning to address class imbalance. SMOTE generates synthetic samples by interpolating between existing minority class samples, effectively increasing the size of the minority class.
- The two key parameters in SMOTE are:
- sampling_strategy: This parameter specifies the desired ratio of the number of samples in the minority class to the majority class after oversampling. The default value is "auto", which sets the minority class size to be equal to the majority class size.
- k_neighbors: This parameter specifies the number of nearest neighbors to consider when generating synthetic samples. 5
- The sampling_strategy parameter can be set to a float value between 0 and 1, indicating the desired ratio of the minority class size to the majority class size. minority
- It is important to choose the right combination of these parameters based on the specific characteristics of the dataset and the machine learning algorithm being used. A common approach is to perform a grid search over different combinations of these parameters to identify the best performing model.



GBM

- n.trees: This parameter controls the number of trees to grow. A higher value of n.trees usually results in better accuracy, but also takes longer to train. Common values range from 50 to 1000. For example: n.trees = 500.

- interaction.depth: This parameter controls the maximum depth of each tree. Increasing this value allows the model to capture more complex relationships in the data, but also increases the risk of overfitting. Common values range from 1 to 10. For example: interaction.depth = 5.

- shrinkage: This parameter controls the learning rate of the algorithm. A lower value of shrinkage usually results in better stability, but also takes longer to train. Common values range from 0.001 to 0.1. For example: shrinkage = 0.01.

- distribution: This parameter sets the distribution of the response variable. For regression problems, the distribution is usually set to "gaussian". For binary classification problems, it is set to "bernoulli". For example: distribution = "gaussian".

- bag.fraction: This parameter controls the fraction of observations used to grow each tree. A lower value of bag.fraction usually results in better robustness, but also reduces the accuracy. Common values range from 0.5 to 0.8. For example: bag.fraction = 0.7.

- train.fraction: This parameter controls the fraction of observations used for training. The remaining observations are used for validation. A higher value of train.fraction usually results in better accuracy, but also increases the risk of overfitting. Common values range from 0.5 to 0.8. For example: train.fraction = 0.7.