Proceedings of the INMM ESARDA 2023 Joint Annual Meeting

Application of machine learning with data encoding techniques to predict stratum-level DP

Lohith Annadevula[1], S. K. Aghara[1]

[1]University of Massachusetts Lowell, Lowell, MA, USA

ABSTRACT

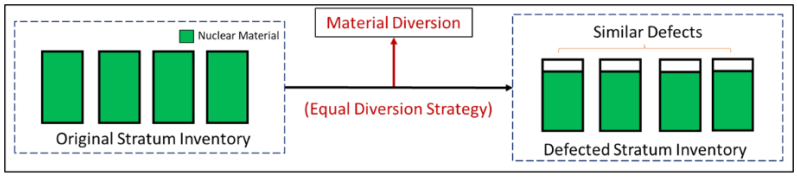
The International Atomic Energy Agency (IAEA) employs well-established statistical methods to assess the effectiveness of its inspection plans on a multi-defect stratum by evaluating defect detection probability (DP). DP is defined as the chance of identifying at least one defect when a defective stratum is subjected to a specific inspection plan. So far, deterministic methods using statistical distributions and a stochastic method using pseudo-random generators have been developed to compute DP within some finite time. The stochastic method is universally applicable to any inspection scenario, and it can generate DP results with user-specified standard error. Initial attempts were made to train machine learning (ML) models on the stochastic DP results and their respective inspection parameters to predict DP. Inspection parameters like item types, instrument types, and identification probabilities vary in length depending on the applied diversion strategy and inspection plan. These variable length parameters pose a major challenge in developing ML models, which require a fixed number of input parameters for training and prediction. The paper explores two ways to convert variable-length parameters to a fixed number of parameters; these are zero-padding and encoding techniques. Zero-padding limits the applicability of models to a few inspection scenarios limiting the variable length parameters to a fixed length, and zeros are used for missing information. On the other hand, Encoding techniques do not limit the model applicability; instead, perform certain operations on the variable length parameters to generate new encoded data with fixed parameters that are used to train ML models. The paper discusses the zero-padding scheme and two different data encoding techniques and compares the performances of ML models trained on said techniques. The R2 scores of zero-padded models and encoded models are evaluated on unseen instances of the test dataset. Upon comparison, show the superior generalization power of encoded models over zero-padded models in predicting DP.

INTRODUCTION

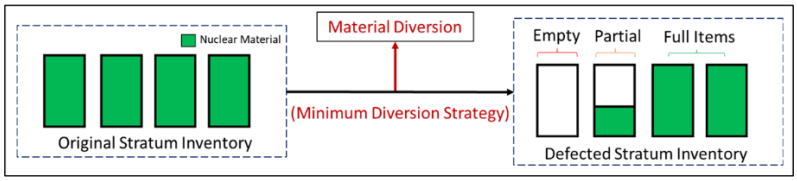
The IAEA's comprehensive safeguards agreement [1] (CSA) obliges a nuclear state or country to subject its nuclear materials inventory to IAEA safeguards operations. The safeguards operations allow IAEA to achieve its technical objective [2] of detecting and deterring nuclear material diversion from peaceful purposes to military use. The nuclear inventory of a state is spread among its nuclear facilities and is reported to IAEA by the state, organized as nuclear strata with items/batches based on similar characteristics like material type, physical state, etc. As part of safeguards operations, IAEA uses a combination of nuclear material accountancy data with surveillance, seal checks, and random inspections to ensure the reported material matches the material present in the strata. Owing to the impracticality of inspecting every item in a stratum, random inspections of a fixed number of items (sample size) are carried out instead based on a specified inspection plan. These inspections involve randomly selecting a specified number of items from the stratum and identifying defects (items from which material has been removed) in the selected items using the instruments and

methods specified in the inspection plan. Each inspection plan has a certain chance of identifying the defects in the stratum called defect detection probability (DP).

The DP is defined as the probability of identifying and detecting at least one defective item when a defected stratum is subjected to a specific inspection plan. DP acts as an effectiveness metric of the IAEA inspection plans and allows IAEA to develop and optimize their inspection plans depending on the anticipated diversion strategies and probable pathways. The diversion strategy is defined as the mechanism by which the proliferator would divert a certain amount of nuclear material (SQ) from the items within a stratum, thereby introducing defective items within the stratum. Depending on the applied diversion strategy, the original items in the stratum inventory are converted into defects of various types and numbers. The IAEA employs models to evaluate the probability of detecting these defects (DP) when an inspection plan is applied to the defected stratum. Figure 1 contains two limiting case diversion strategies that will be used in this paper.



(a) Equal Diversion Strategy



(b) Minimum Diversion Strategy

Figure 1. IAEA limiting case diversion strategies

The IAEA's random inspection process is modeled as random selection and defect identification stages. In the random selection stage, there is a chance of selecting at least one defective item when a certain number of items are randomly sampled from the entire stratum. This chance is called defect selection probability (SP). Its value increases with the number of defects in the stratum or increases with the number of randomly selected items (sample size) from the stratum. In the defect identification stage, the selected items or item is subjected to instrument measurements. There is a chance associated with identifying the measured item as a defect called defect identification probability (IP). IP depends on the instrument or method's response curve and the associated parametric uncertainties like relative standard deviations (RSDs). Hence, the defect detection probability DP is a function of defect selection probability SP and defect identification probability IP. In safeguards literature, the evaluation of DP is traditionally achieved case-by-case using statistical distributions [3]. Recently, two universal DP models were developed, one deterministic and one stochastic model, that can evaluate DP for any inspection scenario and diversion strategy. The deterministic model [4] evaluates DP, equation (1), for each probable out determined using the conditional tree diagram. Whereas the stochastic model [5] uses pseudo-random generators and equation (2) to evaluate DP.

Deterministically,

$$DP = \sum_{i \in \textit{All Selection Outcomes}} \boldsymbol{SP_i} * \boldsymbol{IP_i} \qquad (1)$$

Stochastically, $$\mathrm{DP} = \frac{\mathbf{1}}{\boldsymbol{N}} \sum_{\boldsymbol{i} \in \boldsymbol{Simulated\ Outcomes}} \boldsymbol{IP_i} \quad (2)$$

In existing safeguards literature, the usage of machine learning in predicting DP hasn't been explored before. The main reasons for exploring machine learning for DP prediction are that ML prediction models tend to be faster than deterministic or stochastic models. Once trained, an ML model with reliable accuracy can be used in mobile apps to provide field inspectors with faster and reliable predictions. The paper describes the initial attempts, associated problems, and solutions to predict DP using various machine-learning regression techniques.

OVERVIEW OF ML CONCEPTS

The stratum-level detection probability (DP) is a continuous-valued probability function with values ranging from 0 to 1. The ML methods that predict a continuous value based on an input or multiple input parameters fall under Regression methods. Table 1 contains descriptions of various regression ML methods available as open-source classes in Python's sci-kit learn [6] and xgboost [7] modules.

Table 1. Various regression-based ML methods from sci-kit learn and xgboost modules

S.No	ML Methods	Description	Explored Hyperparameters
1.	Ordinary Least Squares method (OLS)	OLS method is a global minimum method that will provide a unique set of model parameters obtained by minimizing the sum of squares of residuals.	N/A
2.	Linear Non-Linear methods (LNL)	In linear/non-linear methods, the input parameters (x_i) are directly mapped to output parameter y as follows: $y = w_0 + \sum w_i f(x_i)$ Linear or Identity: Linear bottleneck f(x) = x Logistic: the logistic sigmoid function f(x) = 1/(1+exp(-x)) Tanh: the hyperbolic tan function f(x) = tanh(x) Relu: the rectified linear unit function f(x) = max(0, x)	N/A
3.	Deep Neural Network (DNN)	In DNNs, the input layer (x_i) is mapped to the output layer y via multiple (at least two) hidden layers of interconnected neurons. The hidden layer neurons use relu activation, and the final layer uses identity activation.	Hidden layers and neurons: [(), (17), (17, 17), (17, 17, 17), (34), (34, 34), (34, 34, 34), (51), (51, 51), (51, 51, 51), (68), (68, 68), (68, 68, 68), (85,85, 85, 85)]
4.	Support Vector Regression (SVR)	SVR aims to find a hyperplane that best fits the observed data within user-specified residual parameters such as allowed error margin 'e' and tolerance 'C' to values outside margins.	Kernel: ['linear', 'rbf', 'poly'] C: [1, 1.5, 10] Gamma: [1e-7, 1e-10] Epsilon: [0.01, 0.5, 10]
5.	Gradient Boosting Regression (GBR)	GBR is a tree-based boosting ensemble technique that iteratively builds new tree estimators to cover the shortcomings of estimators in the previous iteration. A tree can be summarized in simple terms as nested if-else conditions made of input parameters along with an assigned output value.	Learning rate: [0.5, 0.1, 0.05] No of estimators: [100, 300, 500] Max Tree Depth: [3, 6, 9]
6.	Extra Gradient Boosting Regression (XGBR)	XGBR is also a tree-based boosting ensemble technique like GBR, with added features like regularization, tree pruning from the max depth, missing value handling, etc.	Learning rate: [0.5, 0.1, 0.05] No of estimators: [100, 300, 500] Max Tree Depth: [3, 6, 9]

Table 2 describes various metrics to train ML models and evaluate model performances. The SSR, MAE, and RMSE are suited for both training and performance evaluation. It is customary to generate two datasets, each with unique instances. The instances from the first dataset are used for training various ML models and validating their performances during training. The unseen data

instances from the second dataset are then used to test the generalization capabilities of trained models by evaluating their performance metrics.

Table 2. Formulae and description of various ML metrics.

S.No	Metric	Formulae and Description
1.	Residual	$R_i = y_i - \hat{y}_i$
2.	Sum of Squares Residual (SSR)	$SSR = \sum_{i=1}^{N}(y_i - \hat{y}_i)^2$ SSR takes values from 0 to +∞. The smaller the value of the SSR, the better the model's performance.
3.	Mean Absolute Error (MAE)	$MAE = \frac{\sum_{i=1}^{N}\lvert y_i - \hat{y}_i\rvert}{N}$ MAE takes values from 0 to +∞. The smaller the value of the MAE, the better the model's performance.
4.	Root Mean Square Error (RMSE)	$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{N}}$ $RMSE$ takes values from 0 to +∞. The smaller the value of the $RMSE$, the better the model's performance
where, y_i = actual label value of ith instance $\hat{y}_i$ = predicted label value of ith instance		

k-fold Cross-validation (k-fold CV)

During training, any ML method has multiple parameters that can be varied, which might either improve or worsen the performance. The process of varying parameters in search of a better model is called hyper-parametric tuning. Due to the stochastic nature of solvers and the random initial weights and coefficients, during development the resulting model using the same hyper-parameters will differ with varied performances. To quantify and reduce the variance of performance metrics, a data sampling technique called k-fold cross-validation [8] is used. This approach involves randomly dividing the training dataset into k groups of approximately equal size. In the first iteration, the first fold is treated as a validation set, the model is fit on the remaining k − 1 folds, and performance is evaluated on the first fold. Similarly, models are trained and evaluated for each fold, and k performance metrics are quantified and plotted with mean and standard error. This approach allows better quantification and comparison of performances across various ML algorithms. When combined with hyper-parametric tuning, the k-fold CV method provides a mechanism to identify the optimal parameters within the ML algorithm with the highest performance. In general, with the choice of k in k-fold cross-validation, there is a bias-variance trade-off associated with it. It is typical to choose 5 or 10 as the value for k, as these values have been shown empirically to yield performance metrics that suffer *neither from excessively high bias nor very high variance* [8].

EXPLORED METHODOLOGIES AND THEIR WORKINGS

This section explores the application of various machine-learning techniques in predicting the stratum-level DP. Table 3 contains all the parameters or features taken as the inputs by universal DP models [4,5]. The same parameters are used to develop ML DP models to predict DP. Generally, ML models use constant number of inputs parameters. The dimensions of DP input parameters such as measurement type numbers, item type numbers, and identification probabilities vary depending on the applied diversion strategy and inspection plan. This poses a challenge to develop a ML-based DP

models. Two different approaches are considered here to address this difficulty. The first approach uses zero padding to fix the number of input variables, and the models developed using this approach are called zero-padded models. The second approach uses encoding techniques to fix the number of input variables, and the models developed using this approach are called encoded models.

Table 3. Inputs and output parameters for developing ML DP models

S.No	Parameter	Type, Dimensions, Range	Feature/Label
1.	Total Items (N)	int, 1, $[0, \infty)$	Input Feature
2.	Total Measurements (n)	int, 1, [0, N]	Input Feature
3.	Measurement type numbers ($n_1, n_2, \ldots, n_m$)	Integer array of size m, $\forall\, n_i \in [0, n]$ $\sum_{i=1}^{m} n_i = n$	Input Feature
4.	Item Type numbers ($I_1, I_2, \ldots, I_k$)	Integer array of size k, $\forall\, I_i \in [0, N]$ $\sum_{i=1}^{k} I_i = N$	Input Feature
5.	Identification Probability Matrix $IP_{\text{m by k}}$ $\begin{pmatrix} IP_{11} & IP_{12} & \cdots & IP_{1k} \\ IP_{21} & IP_{22} & \cdots & IP_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ IP_{m1} & IP_{m2} & \cdots & IP_{mk} \end{pmatrix}$	Probability matrix of dimensions k by m $\forall\, IP_{ij} \in [0, 1]$	Input Feature
6.	Stratum-level Defect Detection Probability (DP)	Probability value $DP \in [0, 1]$	Output Label

The zero-padding and encoding procedures to convert variable input features of Table 3 into a fixed number of features for DP model development are described next.

Zero-padding

Zero-padding limits the problem to a few scenarios by fixing the length of parameters and using zeros for missing parameters. The number of item types ‘k’ and the number of instrument types ‘m’ vary depending on the inspection scenario. Here we limit the problem to scenarios that yield three item types and three instrument types by fixing ‘k’ and ‘m’ to 3. Table 4 describes the zero-padded parameters obtained by imposing this constraint to the problem. A total of 17 features are evaluated from zero-padded parameters and used as inputs for the ML models to predict DP. During the data generation process, the scenarios with missing item or instrument type entries are replaced with zeros; hence, the name zero-padding.

Table 41. Constraining variable length parameters for developing zero-padded models

S.No	Original parameters	Zero-padded parameters	No. of features
1.	Total Items (N)	Total Items (N)	1
2.	Total Measurements (n)	Total Measurements (n)	1
3.	Measurement type numbers ($n_1, n_2, \ldots, n_m$)	**m is set to 3** Measurement type numbers (n_1, n_2, n_3)	3
4.	Item Type numbers ($I_1, I_2, \ldots, I_k$)	**k is set to 3** Item Type numbers (I_1, I_2, I_3)	3
5.	Identification Probability Matrix	**k =3, m = 3, k x m = 9**	9

	$IP_{m\ by\ k}$ $\begin{pmatrix} IP_{11} & IP_{12} & \cdots & IP_{1k} \\ IP_{21} & IP_{22} & \cdots & IP_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ IP_{m1} & IP_{m2} & \cdots & IP_{mk} \end{pmatrix}$	$IP_{3\ by\ 3}$ $\begin{pmatrix} IP_{11} & IP_{12} & IP_{13} \\ IP_{21} & IP_{22} & IP_{23} \\ IP_{31} & IP_{32} & IP_{33} \end{pmatrix}$	
	Total Number of Features		**17**

Encoding Procedures

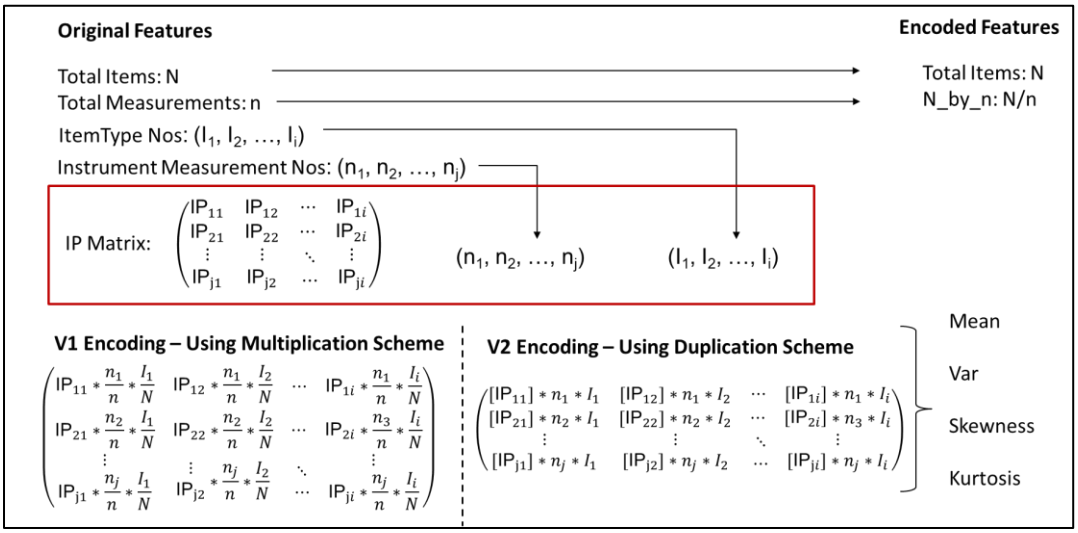


Figure 2. Two different Encoding procedures to convert original parameters into fixed parameters

The zero-padding procedure limits the problem to a few scenarios by fixing the length of parameters and using zeros. For the encoding procedure, the information present in the variable length parameters is encoded to yield a fixed number of parameters. It involves building a distribution of values by combining variable-length parameters and the distribution parameters (mean, variance, etc.) are used as the encoded parameters. The encoding process allows greater flexibility as it does not restrict the number of parameters and hence could be applied to any inspection scenario. Figure 2 describes two different procedures to encode original features to fixed features. The ItemType Nos, Instrument Measurement Nos, and IP Matrix are the variable length features that vary from scenario to scenario. In V1 Encoding, the ItemType Nos array and Instrument Measurement Nos array are normalized to values from 0 to 1 by dividing them by total items (N) and the total measurements (n), respectively. Values are generated by multiplying of normalized itemType Nos array with each column of the IP Matrix followed by multiplication of normalized Instrument Measurement Nos array with each row of the IP Matrix. The distribution matrix is generated. The distribution parameters like mean, variance, skewness, and kurtosis are evaluated, along with total items, and the N_by_n ratio is taken as encoded features to train V1 Encoded ML models. Unlike V1 Encoding, which develops encoding distribution using a multiplication scheme, V2 Encoding uses a duplication scheme to develop the required encoding distribution. Each element of the IP Matrix, say IP_{mk}, is duplicated by the number specified by the ItemTypeNos I_k and MeasurementNos n_m product. The distribution parameters are taken to train V2 Encoded ML models.

RESULTS AND DISCUSSION

Multiple DP models are developed for the six ML methods classes and several hyperparameters described in Table 1. The 10-fold cross-validation technique is used to compare DP model performance on the training data set. For training data set, the OLS method minimizes the sum of squares of residuals (SSR) deterministically, for the remaining ML method classes, mean absolute error (MAE) is minimized using the Adam stochastic gradient solver [11]. For performance comparison, the root mean square error (RMSE) is computed for the DP models using training and testing datasets. The RMSE scores evaluated on the training dataset gives the trained model performance on known instances, whereas the RMSE scores evaluated on the testing dataset give the generalization ability of trained models on unseen data instances. The RMSE metric shares the total error among all instances equally. The smaller the value of RMSE, the better the model performance. Since the DP is a probability function with values between 0 and 1. An RMSE value of 0.8 is interpreted as predicted DP values diverging from actual DP values by a probability of 0.8. For an ML model to make reliable DP predictions, its RMSE score should be less than 0.005 on unseen instances of the test dataset.

Zero-padded Model Results

The inspection scenarios of Table 6, subjected to Figure 1 diversion strategies, are used to generate a training dataset to train zero-padded models of various classes and parameters. One model is selected based on the RMSE score using 10CV training from each class. The criterion for selecting the best model is the lowest RMSE score. The performance (RMSE scores) of the selected models are evaluated using the unseen test dataset. The test dataset is generated from inspection scenarios of Table 7. Both the training and the test RMSE scores are shown in Figure 3.

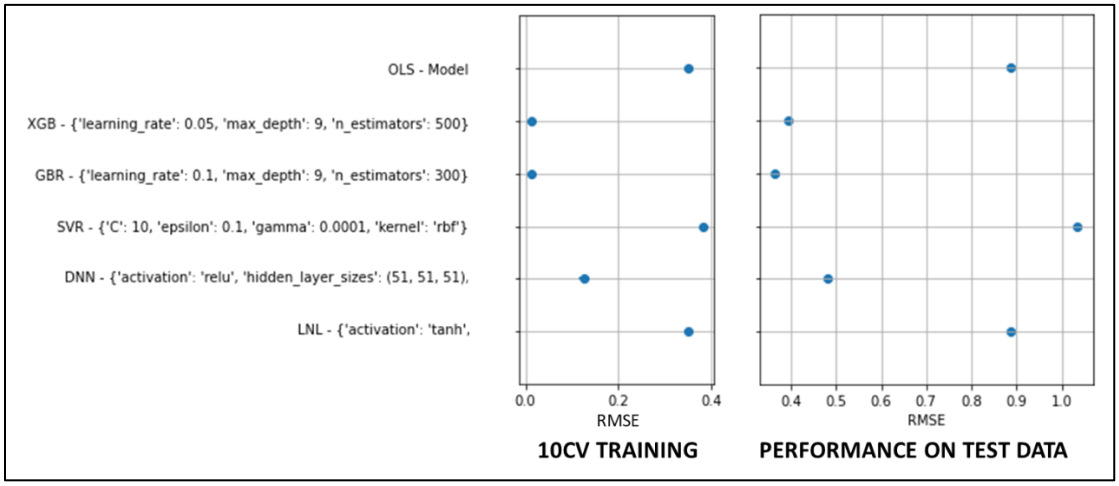


Figure 03. Performance comparison of class-wise best Zero-padded models during 10CV Training and on Test dataset.

Encoded Model Results

The same data instances from the zero-padded training dataset are encoded using procedures of Figure 2 to generate training datasets for encoded-V1 and encoded-V2 models. The inspection scenarios of Table 8, subjected to Figure 1 diversion strategies, are used to generate test datasets for performance evaluation of encoded-V1 and encoded-V2 models. Figures 4 and 5 show the RMSE scores of class-wise best encoded-V1 and encoded-V2 models during 10CV training and their performance on test dataset. A quick comparison of zero-padded and encoded model performance

plots shows that the encoding process lowered RMSE scores for the encoded model results. Indicating improved performance of encoded models over zero-padded model for all classes. Note that both zero-padding and encoding methods used the same training dataset.

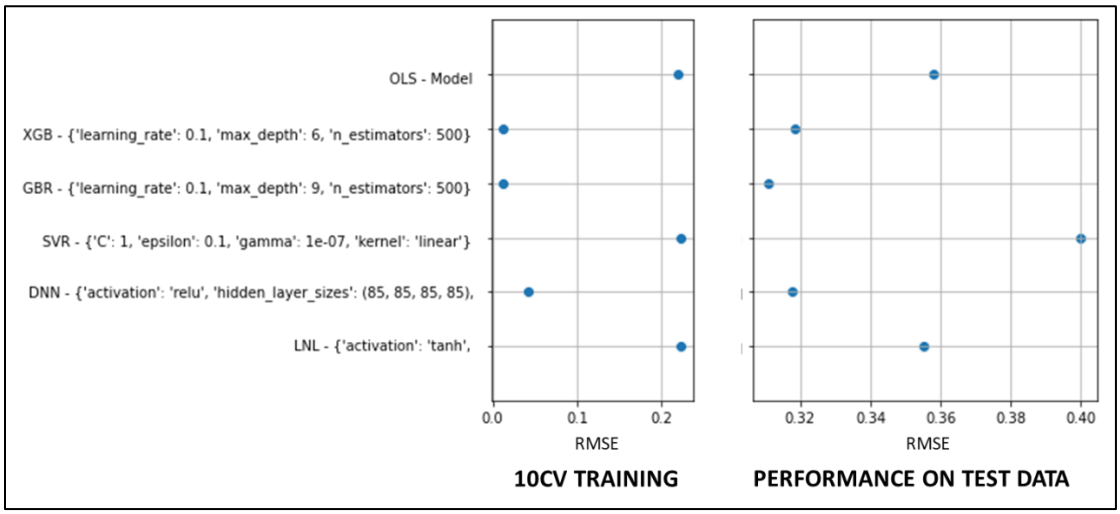


Figure 04. Performance comparison of class-wise best Encoded-V1 models during 10CV Training and on Test dataset.

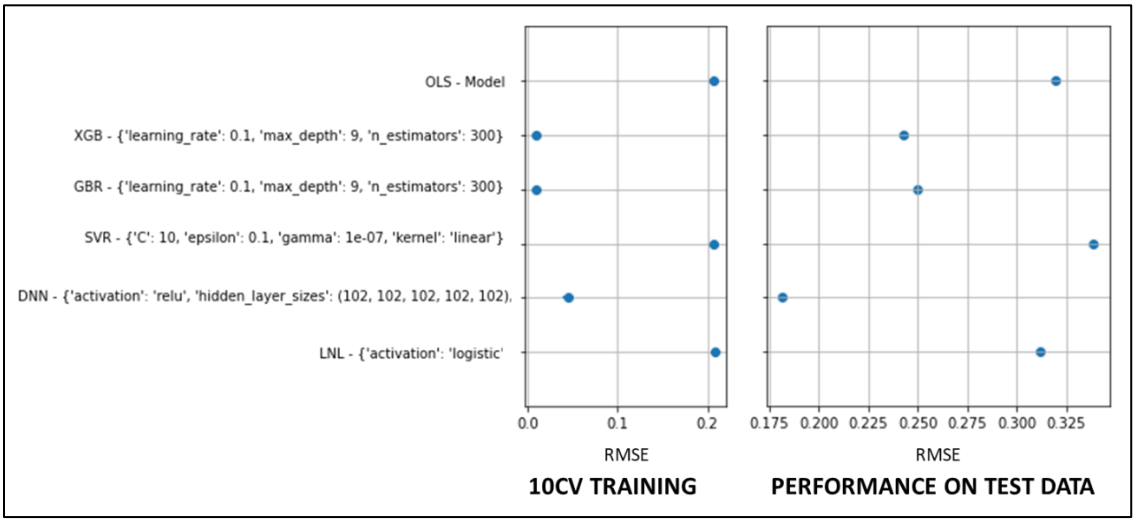


Figure 05. Performance comparison of class-wise best Encoded-V2 models during 10CV Training and on Test dataset.

Table 5 summarizes the performances of best-of-class-wise ML models on the unseen test dataset instances for three different procedures, i.e., zero-padding, encoding v1, and encoding v2. Compared to zero-padding, the encoding process improves the generalizing ability of all ML models on the test dataset. This inference is established by observing the reduction of RMSE scores for every type of ML model between zero-padding and encoding methods. The reduction in RMSE scores from V1 to V2 encoding procedure suggests that further improvements can be achieved with better encoding procedures and parameters.

Table 5. Summary of best model performances (RMSE) on the unseen test dataset

	OLS	LNL	DNN	SVR	GBR	XGBR
Zero-Padded	0.886	0.887	0.481	1.035	0.362	0.392
Encoded V1	0.357	0.355	0.317	0.4	0.31	0.318
Encoded V2	0.31	0.31	0.18	0.34	0.25	0.24

CONCLUSION

The ML models generally use a constant number of input parameters, whereas the DP problem inherently has varying input parameters. The work presented here provided solutions to fix the number of input parameters to develop working ML DP prediction models. Two different procedures, i.e., zero-padding and encoding, are investigated. The ML models are successfully developed for various classes and hyperparameters using both procedures. The results show that by applying encoding procedures, the models produce lower RMSE scores (better results) on both training and test datasets for all classes of ML models investigated in this work. The overall lowest RMSE score of 0.18 on the test dataset was evaluated from the Encoded-V2 DNN class best model. An RMSE score of 0.005 or less is desirable for high confidence in model reliability. Although promising, the value of 0.18 is still much greater than the desired RMSE score. Further work is proposed to develop better encoding procedures and to identify additional distribution parameters like mode and median, which could encompass more information to predict DP. The practical uses of ML-based DP prediction models are found to be limited at this stage, and further investigation is recommended.

Table 6. ML Training dataset scenarios - stratum inventory, inspection plans

Nuclear Inventory									Inspection Plan					
Stratum ID	**Description**	**Items**	**Q%**	**PU (SQ)**	**HEU (SQ)**	**LEU (SQ)**	**NU (SQ)**	**DU (SQ)**	**H**	**F**	**D**	**RSD H**	**RSD F**	**RSD D**
UFE	EU 30B PROD CYL IN STORE	26	0			17.930			1	2	1	0.15	0.05	0.005
UFN	NU 48Y FEED CYL IN STORE	35	0				28.967			3	1		0.07	0.005
UFN-H	NU HEELS CYL IN STORE	40	0				0.009		1			0.15		
UFD	DU 48Y TAILS CYL IN STORE	300	0					121.988	1	1	4	0.15	0.09	0.01
SM1-E	EU SAMPLES AND SLUDGES	163	0			0.015			1			0.15		
UFEP	EU IN PROCESS CYL	9	0			8.299					1			0.005
UFNP	NU IN PROCESS CYL	6	0				2.298				1			0.005
UFDP	DU IN PROCESS CYL	2	0					0.804			1			0.01
FF-1	Fresh Fuel in Dry Store	13728	0			13.176			3			0.15		
FF-1D	Fresh Fuel Dummy in Dry Store	264	0					0.023	1			0.15		
FF-2D	Fresh Fuel Dummy in Pond	264	0					0.023	1			0.15		
SF-	Spent Fuel	223608	0	508.525		38.738			6			0.15		
SR-	Pins in closed container	8	0	0.024		0.004			1			0.15		
UF-	UF6 CYLINDERS	101	0			71.000			1	4	1	0.25	0.05	0.005
PD1	UO2 & U3O8 Powders in Containers	2100	0			30.000			2	3	1	0.25	0.045	0.005
PD2	UO2 & U3O8 Powder in Hoppers	25	0			7.800			1		1	0.25		0.004
PL1	Scintered Pellets in Cans	2250	0			8.500			1		1	0.25		0.003
PL2	Scintered Pellets in Racks	84	0			9.000					2			0.003
FR1	Finished Fuel Rods	19640	0			23.000			3	3		0.25	0.026	
FF-	Finished Assemblies	106	0			27.000			2	4		0.25	0.057	
SC1	CLEAN SCRAP	4540	0			17.800			1	2	1	0.25	0.08	0.04
SD1	DIRTY SCRAP	2910	0			2.100				1			0.14	
FFH	Fresh MTR elements	1280	100		0.800				2			0.05		
FRH	Fresh Moly targets	2000	0		0.380				1			0.05		
CFH	Core MTR Fuel	480	100		0.300				3			0.05		
SFH	Spent MTR Fuel	4800	100		1.050				5			0.05		
UFE1	N = 26; n = 1	26	0			17.930			1			0.05		
UFE1	N = 26; n = 2	26	0			17.930			24			0.05		
UFE1	N = 26; n = 4	26	0			17.930			4			0.05		
UFE2	N = 26; n = 6	26	0			17.930			6			0.05		
UFE3	N = 26; n = 8	26	0			17.930			8			0.05		
UFE4	N = 26; n = 10	26	0			17.930			10			0.05		
UFE5	N = 26; n = 12	26	0			17.930			12			0.05		
UFE6	N = 26; n = 14	26	0			17.930			14			0.05		
UFE7	N = 26; n = 16	26	0			17.930			16			0.05		
UFE8	N = 26; n = 18	26	0			17.930			18			0.05		
UFE9	N = 26; n = 20	26	0			17.930			20			0.05		

UFE10	N = 26; n = 22	26	0			17.930			22			0.05		
UFE11	N = 26; n = 24	26	0			17.930			24			0.05		
UFE12	N = 26; n = 26	26	0			17.930			26			0.05		
UFE13	N = 200; n = 190	200	0			80.000			190			0.05		

Table 7. ML Testing dataset scenarios for evaluating the performance of Zero-padded models

Nuclear Inventory				Inspection Plan					
Stratum ID	**Description**	**Items**	**LEU (SQ)**	**H**	**F**	**D**	**RSD H**	**RSD F**	**RSD D**
U200_n5_1	N = 200; n = 5	200	50.000	5			0.05		
U200_n5_2	N = 200; n = [4 5]	200	50.000	4	1		0.05	0.005	
U200_n5_3	N = 200; n = [3 1 1]	200	50.000	3	1	1	0.05	0.005	0.15
U200_n4_3	N = 200; n = [2 1 1]	200	50.000	2	1	1	0.05	0.005	0.15
U200_n3_3	N = 200; n = [1 1 1]	200	50.000	1	1	1	0.05	0.005	0.15
U5000_n25_1	N = 5000; n = 25	5000	100.000	25			0.05		
U5000_n25_2	N = 5000; n = [20 5]	5000	100.000	20	5		0.05	0.005	
U5000_n25_3	N = 5000; n = [15 5 5]	5000	100.000	15	5	5	0.05	0.005	0.15
U5000_n20_3	N = 5000; n = [10 5 5]	5000	100.000	10	5	5	0.05	0.005	0.15
U5000_n15_3	N = 5000; n = [5 5 5]	5000	100.000	5	5	5	0.05	0.005	0.15

Table 8. ML Testing dataset scenarios for evaluating the performance of Encoded models

Nuclear Inventory				Inspection Plan									
Stratum ID	**Description**	**Items**	**LEU (SQ)**	**H**	**F**	**D**	**G**	**K**	**RSD H**	**RSD F**	**RSD D**	**RSD G**	**RSD K**
U200_n5_1	N = 200; n = 5	200	50.000	5					0.05				
U200_n5_2	N = 200; n = [4 5]	200	50.000	4	1				0.05	0.005			
U200_n5_3	N = 200; n = [3 1 1]	200	50.000	3	1	1			0.05	0.005	0.15		
U200_n5_4	N = 200; n = [2 1 1 1]	200	50.000	2	1	1	1		0.05	0.005	0.15	0.1	
U200_n5_5	N = 200; n = [1 1 1 1 1]	200	50.000	1	1	1	1	1	0.05	0.005	0.15	0.1	0.25
U5000_n25_1	N = 5000; n = 25	5000	100.000	25					0.05				
U5000_n25_2	N = 5000; n = [20 5]	5000	100.000	20	5				0.05	0.005			
U5000_n25_3	N = 5000; n = [15 5 5]	5000	100.000	15	5	5			0.05	0.005	0.15		
U5000_n25_4	N = 5000; n = [10 5 5 5]	5000	100.000	10	5	5	5		0.05	0.005	0.15	0.1	
U5000_n25_5	N = 5000; n = [5 5 5 5 5]	5000	100.000	5	5	5	5	5	0.05	0.005	0.15	0.1	0.25

REFERENCES

1. IAEA, “The Structure and Content of Agreements Between the Agency and States Required in Connection with the Treaty on the Non-Proliferation of Nuclear Weapons,” (INFCIRC/153, Corrected), Vienna: IAEA, 1972.
2. International Atomic Energy Agency; IAEA Safeguards Glossary, International Nuclear Verification Series No. 3; 2003.
3. IAEA, “Statistical Concepts and Techniques for IAEA Safeguards,” IAEA SG-PR-2016 Rev. 5, 1998.
4. Lohith Annadevula, S. K. Aghara; Universal deterministic modeling to compute stratum-level detection probability based on conditional tree diagram, submitted to Proc. of the INMM 63rd Annual Meeting; 2022.
5. Lohith Annadevula, S.K. Aghara, Kenneth Jarman and Logan Joyce; Stochastic Approach to Inspection Evaluation: Methodology and Validation. ESARDA Bulletin - The International Journal of Nuclear Safeguards and Non-proliferation, 64(1), 30-38. (2022, June). https://doi.org/10.3011/ESARDA.IJNSNP.2022.3
6. F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.
7. T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794, 2016, doi: 10.1145/2939672.2939785.
8. G. James, D. Witten, T. Hastie, and R. Tibshirani, "k-Fold Cross-Validation," in An introduction to statistical learning : with applications in R: New York : Springer, 2013, pp. 181-183.