STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **14**, December 2025, pp 3804–3815.
Published online in International Academic Press (www.IAPress.org)

A Computational Study for Probabilistic Fuzzy Linear Programming Using Machine Learning in the Case of Poisson Distribution

Maged George Iskander[1], Israa Lewaaelhamd[2,*]

[1]*Department of Mathematics and Actuarial Science, The American University in Cairo, New Cairo 11835, Egypt*
[2]*Department of Business Administration, Faculty of Business Administration, The British University in Egypt, Cairo, Egypt*

Abstract This paper presents a computational study for probabilistic fuzzy linear programming using machine learning. Two opposite probabilistic fuzzy constraints are considered, where the random variable in the two constraints is discrete with a Poisson distribution. The data was generated from a Poisson distribution under different scenarios. Five scenarios are investigated based on either the same mean parameter (Poisson distribution parameter) and different dispersions of the values of the random variable or the same values of the random variable and different mean parameters. Moreover, eight cases are derived by considering different combinations of fuzzy probabilities. These many configurations of different scenarios and cases allow us to compare how the model performs while varying both the mean parameter and the range of the values through different combinations of fuzzy probabilities. This setup allows for a thorough evaluation of how these changes impact model performance using machine learning models. Nine machine learning models have been considered in this study for evaluating different scenarios and cases in predicting the target decision variables. Since the Poisson distribution is beneficial in fields such as telecommunications, healthcare, logistics, and reliability engineering, where the frequency of arrivals, failures, or demands exhibits Poisson-like behavior but is additionally impacted by ambiguity or incomplete information. Therefore, this study provides a useful tool to the decision-makers to carefully select the combinations of the fuzzy probabilities in the light of the possible values of the Poisson random variable, especially when the associated probabilities are not specified.

Keywords Probabilistic fuzzy linear programming, Simulation study, Machine learning models, Poisson distribution.

AMS 2010 subject classifications 60E05, 62E10, 62F10.

DOI: 10.19139/soic-2310-5070-2800

1. Introduction

Fuzzy Linear Programming (FLP) plays an important role in dealing with uncertainty in decision-making problems. By integrating fuzzy set theory with linear programming, FLP offers a way to deal with imprecision and lack of information in model parameters. One of the first and most significant methods was introduced by [36], where the constraints and objectives are fuzzy in the optimization framework. This allows the decision-makers to work with approximate values rather than solely depending on precise values. This strategy has proven particularly helpful in controlling uncertainty ([31];[15]; [18];[11];[10]; [6];[33]).

Uncertainty can arise from different sources, involving both randomness and fuzziness. Fuzzy stochastic programming models have been developed by several studies ([21];[1];[22];[26];[14]). As a result, Probabilistic Fuzzy Linear Programming (PFLP) models were developed, which integrate fuzzy logic and probabilistic methods like chance-constrained to capture complex uncertainty effectively. [2] proposed fuzzy stochastic programming problems with a crisp objective function and linear constraints whose coefficients are fuzzy random variables of type L-R. To address these difficulties, deterministic counterparts of chance-constrained programming with fuzzy stochastic coefficients are developed by incorporating probability, possibility, and necessity constraints. Recently, [17] tackled multi-objective decision-making under uncertainty by developing a nonlinear fuzzy chance-constrained framework within a mixed fuzzy stochastic programming approach.

*Correspondence to: israa.lewaa@bue.edu.eg; israalewaa@feps.edu.eg

ISSN 2310-5070 (online) ISSN 2311-004X (print)

They proposed a new solution named fuzzy stochastic Pareto optimal solution. The efficiency of this solution is demonstrated through numerical comparisons with existing methods.

[12] proposed a novel method for solving stochastic fuzzy linear programming problems where random variables in both constraints and the objective function are discrete with triangular fuzzy probabilities. His approach applies the α-cut method to fuzzy probabilities and uses chance-constrained programming under strict or general dominance relations for constraints. The objective function incorporates the mean-variance criterion for the general case and is demonstrated by a numerical example. In a later study, [13] extends his approach to a general multi-objective programming model with imprecise probabilities. Both the right-hand side coefficients of the constraints as well as the coefficients in the objective functions are considered independent discrete random variables. The methodology involves three steps: optimizing the α-cut value to determine the best probability levels, identifying desirable objective function values, and applying the mean-variance method within a global criterion framework.[32] focuses on multi-objective two-level simple recourse programming problems with discrete-type LR fuzzy random variables. This paper introduces a new solution concept called an estimated Pareto Stackelberg solution. Then, an interactive algorithm to obtain a satisfactory solution from among an estimated Pareto Stackelberg solution set was proposed. A numerical example illustrated the proposed algorithm for a multi-objective two-level fuzzy random simple recourse programming problem.

Dealing with count data, where events occur randomly over time or space, is the primary difficulty in many practical applications addressed by Probabilistic Fuzzy Linear Programming (PFLP). This type of count data may naturally be modeled by a discrete probability distribution such as the Poisson distribution. Incorporating the Poisson distribution into PFLP frameworks permits more realistic modeling of systems characterized by random event occurrences coupled with uncertainty or imprecision. This is especially pertinent in fields such as telecommunications, healthcare, logistics, and reliability engineering, where the frequency of arrivals, failures, or demands exhibits Poisson-like behavior but is additionally impacted by ambiguity or incomplete information. For instance, [19] introduced a stochastic Poisson process in which the rate parameter is considered as a fuzzy variable. More recently, [25] developed the Double Fuzzy Poisson Distribution to handle two levels of uncertainty commonly encountered in biological data. Their approach proved a robust and adaptable tool for handling biological data characterized by multiple sources of imprecision, making it a valuable addition to the field of biomathematical modeling.

The Poisson distributions and fuzzy or probabilistic linear programming approaches have been effectively used in several studies to address uncertainty in complex systems. For instance, [34] developed a novel multi-stage possibilistic stochastic programming approach that integrates the Poisson distribution with fuzzy parameters, aiming to manage deep uncertainty in early-stage disaster relief planning. Their proposed approach is tested based on a real case study for the post-disaster relief distribution planning, and useful managerial insights are provided through conducting several sensitivity analyses. In another field, [5] introduced a risk-based probabilistic integer programming model designed to optimize blasting costs in surface mining. Their approach utilizes both Poisson and exponential distributions to consider the probability of undesirable blast-induced impacts in the model. In the field of quality control, [8] proposed a quality Interval acceptance single sampling plan when the fraction of nonconforming items is a fuzzy number and is modeled based on the fuzzy Poisson distribution. This study carried out a new procedure for implementing fuzzy logic in quality Interval acceptance sampling plan. Similarly, [7] proposed a mathematical programming method to construct the membership functions of the fuzzy objective value of the cost-based queueing decision problem with the cost coefficients and the arrival rate being fuzzy numbers. In this model, the arrival rate follows a Poisson distribution with fuzzy parameters. Two numerical examples were solved successfully to demonstrate the validity of their proposed method.

Recent developments in optimization have placed greater emphasis on combining fuzzy and probabilistic linear programming with Machine Learning (ML) techniques to enhance decision-making under uncertainty. In an earlier contribution, [9] concentrated on two types of fuzzy linear programming problems. The first type with fuzzy coefficients in the objective function, and the second type with fuzzy right-hand side values and fuzzy variables. In their study, fuzzy neural network models are used to address problems involving fuzzy differential equations and fuzzy derivatives. To show the applicability of the method, it is applied to solve the fuzzy shortest path problem and the fuzzy maximum flow problem. Recently, [16] proposed a methodology based on machine learning and fuzzy logic, enhancing the inventory management in dynamic sectors like the pharmaceutical industry. Their study integrates the Naive Bayes (NB) classifier and the use of the Weka artificial intelligence for increasing the effectiveness of their model in complex decision-making environments. [3] addressed fuzzy multi-objective linear programming problems focusing on applications in smart city construction. The study applied two methods, which are a mathematical model and Artificial Neural Networks (ANNs). Results indicate that the ANN approach offers a notable time and effort savings when compared to traditional mathematical models. Similarly, [20] developed a Deep Reinforcement Learning approach (DRL) guided by fuzzy logic and designed to improve the agent's learning efficiency in identifying optimal strategies.

The rest of the paper is organized as follows: Section 2 provides the probabilistic fuzzy linear programming model and its deterministic crisp equivalent. In Section 3, the proposed scenarios and cases of fuzzy probabilities are elaborated. In Section 4, the results of the computational study using machine learning models are presented to get useful insights. Finally, Section 5 presents conclusions and future research directions.

2. The probabilistic fuzzy linear programming model and its deterministic crisp equivalent

Assume the following two probabilistic fuzzy constraints:

$$\Pr(x \geq b) \gtrsim \beta_1, \tag{1}$$
$$\Pr(x \leq b) \gtrsim \beta_2, \tag{2}$$

where x is a non-negative decision variable, and β_1 is the required probability level for constraint (1), while β_2 is the required probability level for constraint (2), $0 < \beta_i < 1$ $(i = 1, 2)$. Also, b is a discrete random variable having a Poisson distribution, and $\gtrsim$ is the fuzziness of $\geq$. Let k be the number of values of the random variable b whose possible positive values are $y_1, y_2, \ldots, y_k$, which have an ascending order and are, respectively, associated with the probabilities $p_1, p_2, \ldots, p_k$, where $\sum_{l=1}^{k} p_l = 1$. In addition, for each of the two probabilistic fuzzy constraints, δ_i is incorporated as a predetermined value representing the lowest allowed probability level, $0 < \delta_i < \beta_i$ $(i = 1, 2)$. Therefore, the membership function for the probabilistic fuzzy constraint (1) can be presented as

$$\mu_1(x) = \begin{cases} 1 & \text{if } \Pr(x \geq b) \geq \beta_1, \\ \dfrac{\Pr(x \geq b) - \delta_1}{\beta_1 - \delta_1} & \text{if } \delta_1 \leq \Pr(x \geq b) \leq \beta_1, \\ 0 & \text{if } \Pr(x \geq b) \leq \delta_1, \end{cases} \tag{3}$$

while for (2) it is defined by

$$\mu_2(x) = \begin{cases} 1 & \text{if } \Pr(x \leq b) \geq \beta_2, \\ \dfrac{\Pr(x \leq b) - \delta_2}{\beta_2 - \delta_2} & \text{if } \delta_2 \leq \Pr(x \leq b) \leq \beta_2, \\ 0 & \text{if } \Pr(x \leq b) \leq \delta_2. \end{cases} \tag{4}$$

By utilizing the α-cut approach for the two membership functions, the probabilistic fuzzy constraints (1) and (2) are, respectively, stated as

$$\Pr(x \geq b) \geq \alpha\beta_1 + (1 - \alpha)\delta_1, \tag{5}$$

and

$$\Pr(x \leq b) \geq \alpha\beta_2 + (1 - \alpha)\delta_2, \tag{6}$$

where $\alpha \in [0, 1]$, which is required to be maximized. Therefore, using the chance-constrained approach for (5) and (6), the equivalent constraints can, respectively, be given by

$$x \geq \min \{y_l : F(y_l) \geq \alpha\beta_1 + (1 - \alpha)\delta_1,\ l = 1, 2, \ldots, k\}, \tag{7}$$

and

$$x \leq \max \{y_l : F(y_{l-1}) \leq 1 - (\alpha\beta_2 + (1 - \alpha)\delta_2),\ F(y_0) = 0,\ l = 1, 2, \ldots, k\}, \tag{8}$$

where $F(.)$ is the cumulative distribution function of the Poisson random variable b. Therefore, since $y_1, y_2, \ldots, y_k$, are arranged according to an ascending order, then (7) and (8) can be presented by the following set of constraints:

$$\sum_{l=1}^{k} q_{1l} \sum_{t=1}^{l} p_t \geq \alpha\beta_1 + (1 - \alpha)\delta_1, \tag{9}$$

$$x \geq \sum_{l=1}^{k} q_{1l} y_l, \tag{10}$$

$$\sum_{l=1}^{k-1} q_{2l+1} \sum_{t=1}^{l} p_t \leq 1 - (\alpha\beta_2 + (1-\alpha)\delta_2), \tag{11}$$

$$x \leq \sum_{l=1}^{k} q_{2l} y_l, \tag{12}$$

$$\sum_{l=1}^{k} q_{jl} = 1, \quad j = 1, 2, \tag{13}$$

$$q_{jl} \in \{0, 1\}, \quad j = 1, 2;\ l = 1, 2, \ldots, k, \tag{14}$$

$$0 \leq \alpha \leq 1, \tag{15}$$

$$x \geq 0. \tag{16}$$

Hence, the deterministic crisp model is presented as

Maximize α

Subject to: (9)–(16), (17)

where α and x are the relevant decision variables.

Proposition 1

Constraint (11) can be represented by the following constraint:

$$\sum_{l=1}^{k} q_{2l} \sum_{t=l}^{k} p_t \geq \alpha\beta_2 + (1-\alpha)\delta_2. \tag{18}$$

Therefore, model (17) may be replaced with the following equivalent model:

Maximize α

Subject to: (9)–(10), (18), (12)–(16). (19)

It is obvious that both models (17) and (19) take the form of mixed zero-one linear programs.

Proposition 2

According to model (17) or (19), if the optimal value of α (α^*) exists, then decreasing the value of β_1 and/or β_2 while keeping the values of δ_1 and δ_2 unchanged or decreasing the value of δ_1 and/or δ_2 while maintaining the values of β_1 and β_2 unchanged, should result to an optimal value of α that is greater than or equal to α^*. The results of the computational study were consistent with Proposition 2, confirming the theoretical expectation.

The decision-maker should be careful in setting his/her probabilities to avoid infeasible and unsatisfactory solutions. For instance, Iskander provided a normalization for the decision-maker's required probabilities [12]. According to Iskander's proposition, in the case of the strict dominance relation, the α-cut value should satisfy the condition that the normalized required probabilities must lie between zero and one inclusive. Hence, for any probabilistic fuzzy constraint, both the required probability and all the lowest values of the fuzzy probabilities should not equal zero at the same time. That is to avoid getting an indeterminate form of the corresponding normalized required probability when the chosen value of the α-cut is zero as well. In general, the required probability should not practically equal zero, and, in any case, if all the lowest values of the fuzzy probabilities are equal to zero, then the α-cut value must not be zero.

3. The proposed scenarios of the Poisson distribution

3.1. Scenario design

To investigate the effect of different values of the Poisson distribution parameter, i.e., mean parameter (λ) and y-value ranges, five scenarios were developed, as shown in Table 1. Scenarios 1, 2, and 3 all fix λ at 50 but vary in the spread of y-values. Scenario 1 includes values close to the mean ($y = 48, 49, 50, 51, 52$), Scenario 2 expands the range moderately ($y = 30, 40, 50, 60, 70$), and Scenario 3 explores a much wider and more extreme spread ($y = 10, 30, 50, 70, 90$). In Scenario 4, the y-values remain the same as in Scenario 2, but λ is reduced to 40. Conversely, Scenario 5 keeps the y-values from Scenario 3 while increasing λ to 60. These variations allow us to compare how models perform while considering different y-value ranges under the same λ and to examine the impact of changing λ while keeping the y-values constant. By varying both λ and the range of y-values through different scenarios, this setup allows for a thorough evaluation of how changes in data distribution impact model performance. The five scenarios are summarized in Table 1.

Table 1. Different scenarios of λ and corresponding y-values

Scenario	Description
Scenario 1	$\lambda = 50$: y-values = [48, 49, 50, 51, 52].
Scenario 2	$\lambda = 50$: y-values = [30, 40, 50, 60, 70].
Scenario 3	$\lambda = 50$: y-values = [10, 30, 50, 70, 90].
Scenario 4	$\lambda = 40$: y-values = [30, 40, 50, 60, 70].
Scenario 5	$\lambda = 60$: y-values = [10, 30, 50, 70, 90].

Therefore, to simulate realistic probabilistic behaviors by using predefined y-values, a data set of the corresponding probabilities was generated from the Poisson distribution. For each scenario, probabilities $\Pr(y_i)$ were assigned to y-value (y_i) such that the expected value, $\sum_i y_i \Pr(y_i)$, is as close as possible to the target value of λ.

To ensure precision, the expected value for each distribution was constrained to lie within a narrow interval around its corresponding λ. For λ = 40: Expected value $\in [39.9, 40.1]$. For λ = 50: Expected value $\in [49.9, 50.1]$. For λ = 60: Expected value $\in [59.9, 60.1]$. A total of 100 distributions were generated per scenario, allowing variability in probability distributions but still staying close to the target values of λ.

3.2. Fuzzy probabilities combinations

The required probabilities β_1 and β_2, as well as the lowest allowable probabilities δ_1 and δ_2 as defined in Section 2, may take different values. The values of β's and δ's were investigated, and it is recommended that $\beta_1, \beta_2 \in \{0.8, 0.9\}$ while $\delta_1, \delta_2 \in \{0.4, 0.6\}$.

These values produce plenty of optimal solutions; hence few infeasible solutions were found and removed. A total of eight main cases were considered. The different combinations of fuzzy probabilities are listed in Table 2.

Table 2. Different cases for β and δ values

Case	Description	Exact Values
Case 1	$\beta_1 = \beta_2$ and $\delta_1 > \delta_2$	Case 1.1: $\beta_1 = 0.9$, $\beta_2 = 0.9$, $\delta_1 = 0.6$, $\delta_2 = 0.4$ Case 1.2: $\beta_1 = 0.8$, $\beta_2 = 0.8$, $\delta_1 = 0.6$, $\delta_2 = 0.4$
Case 2	$\beta_1 = \beta_2$ and $\delta_1 < \delta_2$	Case 2.1: $\beta_1 = 0.9$, $\beta_2 = 0.9$, $\delta_1 = 0.4$, $\delta_2 = 0.6$ Case 2.2: $\beta_1 = 0.8$, $\beta_2 = 0.8$, $\delta_1 = 0.4$, $\delta_2 = 0.6$
Case 3	$\beta_1 > \beta_2$ and $\delta_1 = \delta_2$	Case 3.1: $\beta_1 = 0.9$, $\beta_2 = 0.8$, $\delta_1 = 0.4$, $\delta_2 = 0.4$ Case 3.2: $\beta_1 = 0.9$, $\beta_2 = 0.8$, $\delta_1 = 0.6$, $\delta_2 = 0.6$
Case 4	$\beta_1 < \beta_2$ and $\delta_1 = \delta_2$	Case 4.1: $\beta_1 = 0.8$, $\beta_2 = 0.9$, $\delta_1 = 0.4$, $\delta_2 = 0.4$ Case 4.2: $\beta_1 = 0.8$, $\beta_2 = 0.9$, $\delta_1 = 0.6$, $\delta_2 = 0.6$
Case 5	$\beta_1 > \beta_2$ and $\delta_1 > \delta_2$	$\beta_1 = 0.9$, $\beta_2 = 0.8$, $\delta_1 = 0.6$, $\delta_2 = 0.4$
Case 6	$\beta_1 < \beta_2$ and $\delta_1 < \delta_2$	$\beta_1 = 0.8$, $\beta_2 = 0.9$, $\delta_1 = 0.4$, $\delta_2 = 0.6$
Case 7	$\beta_1 > \beta_2$ and $\delta_1 < \delta_2$	$\beta_1 = 0.9$, $\beta_2 = 0.8$, $\delta_1 = 0.4$, $\delta_2 = 0.6$
Case 8	$\beta_1 < \beta_2$ and $\delta_1 > \delta_2$	$\beta_1 = 0.8$, $\beta_2 = 0.9$, $\delta_1 = 0.6$, $\delta_2 = 0.4$

4. Computational study and machine learning

4.1. Data preparation

To comprehensively evaluate the performance of the model under a variety of conditions, we considered all possible combinations of the five scenarios outlined in Table 1 and the eight main cases detailed in Table 2, resulting in 60 unique datasets. Each dataset contains 100 observations, yielding a total of 6,000 distributions across all combinations. Model (17) has been implemented to 6000 distributions, where the values of the target decision variables x and α, for each distribution, have been recorded. The Python 3.13.3 software was utilized in this computational study. Hence, we applied the machine learning models to predict the target decision variables x and α. Accordingly, we examined model performance from two perspectives: first, a general approach that considers all the scenarios and cases, and second, a detailed analysis that considers each scenario with all cases as well as each case with all scenarios. This dual-level analysis provides a robust framework for understanding how data organization influences predictive accuracy and for evaluating model sensitivity and adaptability under diverse conditions.

Before model training, we cleaned the datasets by unifying the column names and removing rows with infeasible solutions. The input features used in the machine learning models are $p_1, p_2, p_3, p_4, p_5, y_1, y_2, y_3, y_4, y_5, \beta_1, \beta_2, \delta_1$ and δ_2.

4.2. Machine learning models and performance metrics

Nine well-known machine learning models were used in this study, which are Support Vector Regression (SVR), Ridge Regression, Linear Regression, AdaBoost, K-Nearest Neighbors (KNN), Decision Tree, XGBoost, Random Forest, Gradient Boosting ([27]; [28]; [23]; [4]; [35]; [29]; [24]; [30]). The data was split into training and testing data for each machine learning model randomly with 80% and 20% respectively. Models were trained and their performance was evaluated based on the testing data using the R^2 score and Root Mean Square Error (RMSE). Table 3 summarizes the values of R^2 and RMSE for the overall performance in predicting α and x of the nine machine learning models when considering all the scenarios and cases.

Table 3. Performance metrics (R^2 and RMSE) for predicting α and x using nine machine learning models

Machine Learning Models	R^2 for α	RMSE for α	R^2 for x	RMSE for x
SVR	0.0324	0.2401	0.5346	7.8956
Ridge Regression	0.2249	0.2149	0.6169	7.1638
Linear Regression	0.2250	0.2149	0.6171	7.1619
AdaBoost	0.4197	0.1808	0.6688	6.7244
KNN	0.4516	0.1872	0.6624	6.6546
Decision Tree	0.4903	0.1743	0.6707	6.6412
XGBoost	0.4911	0.1744	0.6710	6.6397
Random Forest	0.4914	0.1741	0.6714	6.6384
Gradient Boosting	0.5301	0.1673	0.6991	6.3493

Table 3 shows that predicting α was generally more challenging than predicting x, as all models achieved lower performance for this target. Among them, Gradient Boosting achieved the best results, with an R^2 value of about 0.53, while simpler models such as Ridge and Linear Regression showed considerably weaker performance. In contrast, when predicting x, all models performed better overall. Gradient Boosting again achieved the highest accuracy, with an R^2 of approximately 0.70, followed by Random Forest and XGBoost, which also demonstrated strong predictive capabilities. SVR, however, recorded the lowest R^2 value of around 0.53.

Overall, this comparison highlights that while α is a more difficult variable to predict, advanced ensemble algorithms like Gradient Boosting and Random Forest consistently outperform simpler regression models across both prediction tasks.

4.3. Feature Importance Analysis

To investigate the contribution of each feature used to predict α and x, we applied SHAP (SHapley Additive exPlanations) analysis on the best-performing model, which is the Gradient Boosting model. Figure 1 presents horizontal bar charts of the mean absolute SHAP values for each feature.

For α, the SHAP values are smaller overall, suggesting lower sensitivity. The most impactful features are δ_2, δ_1, and p_3, while the other features, including the β values, have minimal influence.

For x, the most influential features are λ, δ_2, δ_1, and p_4, indicating that both the Poisson parameter λ and the lower bound probabilities δ_1 and δ_2 strongly affect the optimal decision variable. The remaining features, including β_1 and β_2, have relatively smaller contributions.

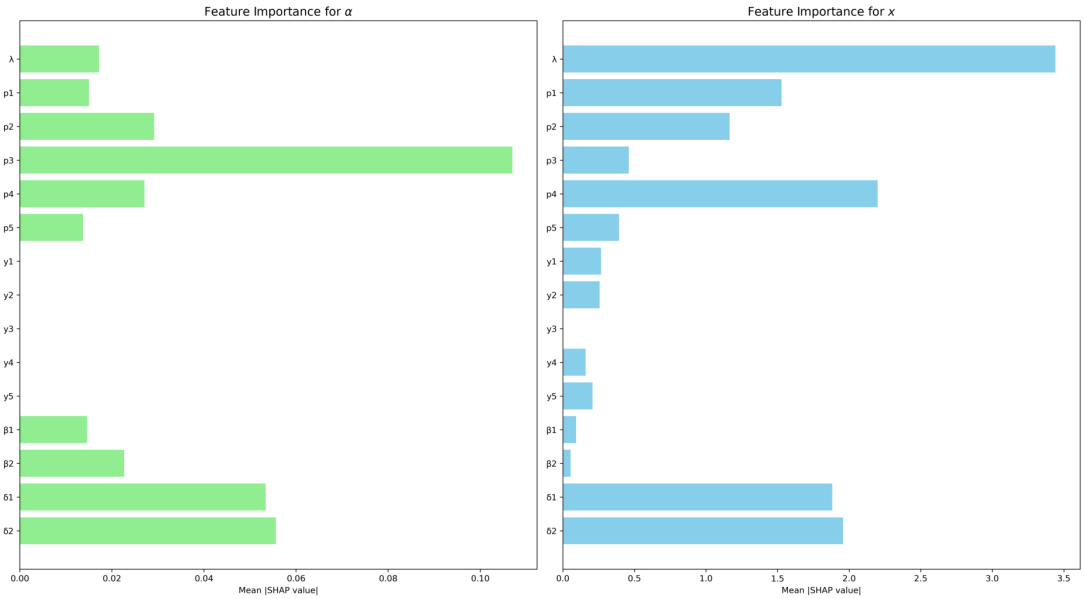

Figure 1. Horizontal bar charts showing mean absolute SHAP values for each feature in the Gradient Boosting model predicting α and x

These results provide visual and quantitative evidence for the relative importance of the input features in determining the outputs of the model α and x, supporting our sensitivity analysis and offering guidance for decision-making in practice.

4.4. Performance of machine learning models across scenarios

For further investigation, we focused on the top four performing machine learning models for each scenario, with all cases as well as each case with all scenarios.: Decision Tree, XGBoost, Random Forest, and Gradient Boosting. These four models consistently showed strong and comparable accuracy in predicting both target variables, x and α. Now, we will consider that each case includes data from all different scenarios, and likewise, each scenario contains data from all the different cases. This ensures a wide variety of data combinations to give us a clearer understanding of their true predictive performance across diverse settings.

Figures 2 and 3 present the R^2 scores of the top four machine learning models for predicting both α and x, Random Forest, Gradient Boosting, Decision Tree, and XGBoost across the five scenarios.

Although both α and x were predicted using the same models, α was generally easier to learn than x across all scenarios. The models gave better results for α, possibly because it is less affected by variations in the y-values and λ in the Poisson data. This might mean that α has a simpler or more stable structure, allowing the models to perform better and make more accurate predictions.

The way y-values are spread around λ clearly affects how well the models can predict x. When the data in Scenario 2 and Scenario 3 were scattered more widely and farther away from λ, the R^2 scores had the lowest values compared to all scenarios. On the other hand, in Scenario 1 and Scenario 4, where the values were closer to λ, the predictions improved. Interestingly, Scenario 5 gave the best prediction results, even though the data had a wide range. This shows that alignment with λ, rather than just the dispersion, plays a major role in prediction accuracy.

In all the scenarios, the models better predict α than x. The performance changed a bit depending on the scenario. The best results were in Scenario 2 and Scenario 3, where the y-values were more widely dispersed around $\lambda = 50$. These scenarios seemed to predict α better when y-values showed more dispersion. However, even though Scenario 5 had a similar dispersion to Scenario 3, changing λ from 50 to 60 caused a drop in performance. Similarly, Scenario 4, which had the same dispersion as Scenario 2, changing λ from 50 to 40, also led to lower performance. This suggests that it's not just the dispersion of the data that matters, but also how that dispersion relates to λ. In other words, α is generally easier for the models to predict than x across different scenarios, but its predictability still depends on how the y-values are spread around λ, which plays a key role in the accuracy of the prediction.

Gradient Boosting is the best prediction model, which achieves the highest R^2 scores across the five scenarios for predicting both x and α. Random Forest and XGBoost followed closely with comparable results. Decision Tree

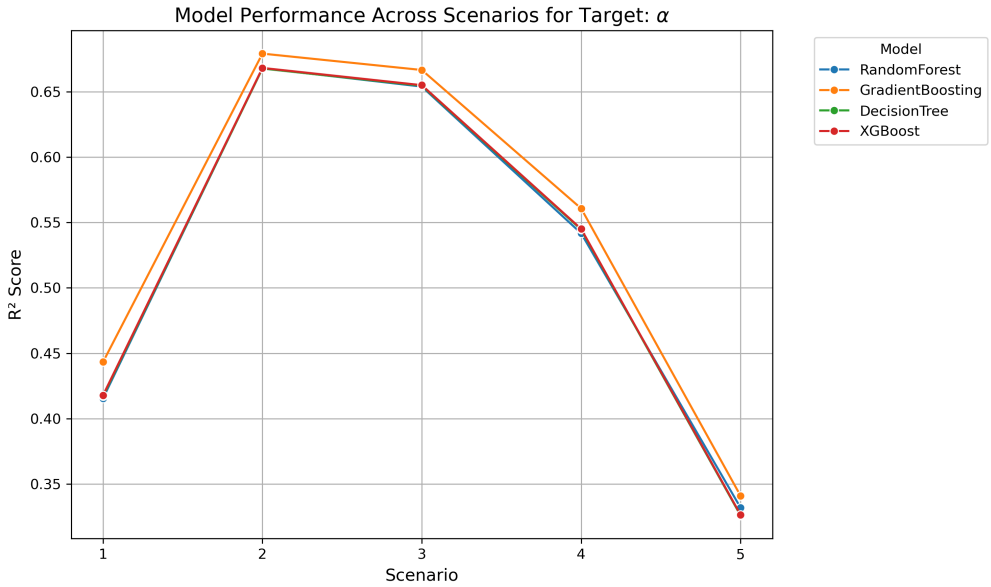

Figure 2. R^2 values of the top four machine learning models across the five scenarios in predicting α

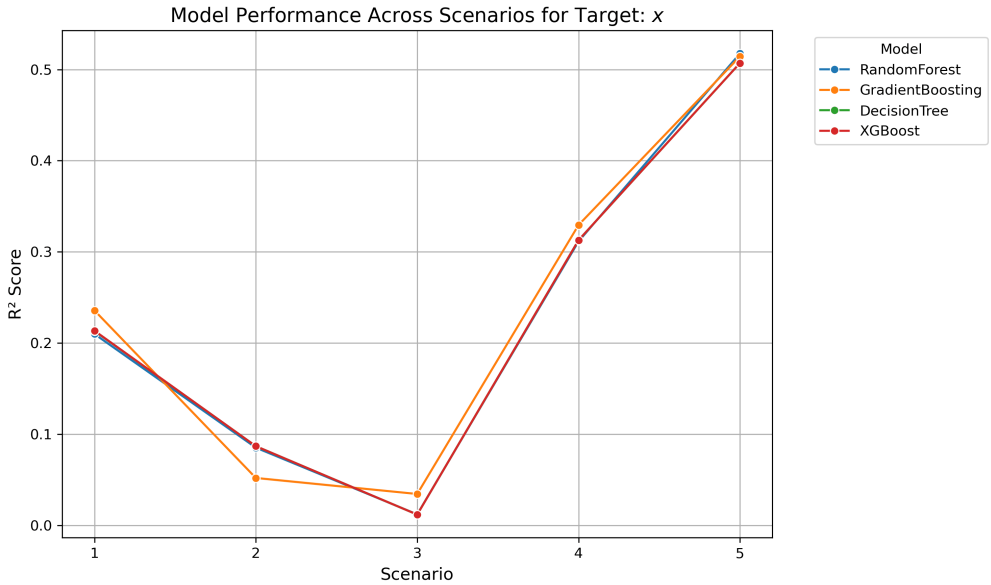

Figure 3. R^2 values of the top four machine learning models across the five scenarios in predicting x

underperformed compared to the other three machine learning models. Although Gradient Boosting achieved the highest accuracy, simpler models like Decision Trees may be more practical when interpretability and transparency are required. Overall, this analysis shows that both the choice of what we are trying to predict and how the data is set up are very important for how well the models work. Predicting values like α, which are more stable, makes it easier for models to learn. Also, choosing y-values that are well aligned with λ leads to better results. These findings are especially useful when working with data that follows a probability distribution, where small changes in data setup can make a big difference in accuracy.

4.5. Performance of machine learning models across cases

Figures 4 and 5 present the R^2 scores of the top four machine learning models for predicting both α and x, which are Random Forest, Gradient Boosting, Decision Tree, and XGBoost across the eight cases, while considering different combinations of β and δ.

Across the eight cases, all machine learning models gave strong results when predicting x, as shown by the high R^2 scores. In most cases, the R^2 values were above 0.90, especially for Random Forest and XGBoost. This means that x is easy to learn under different settings of β and δ. In contrast, the prediction of α varies more across the cases and is strongly influenced by how β and δ interact. The best predictive performance for α was achieved in Case 6, where $\beta_1 < \beta_2$ and $\delta_1 < \delta_2$. In this case, all models, especially Random Forest and XGBoost achieved R^2 values above 0.92. This suggests that having both β_1 and δ_1 at lower levels may make α easier for the model to learn. Similarly, Case 1 and Case 2, where $\beta_1 = \beta_2$ and δ values are not equal, showing relatively strong model performance for α. This suggests that having equal β values can support better predictability. The same happened in Case 8 (where $\beta_1 < \beta_2$ and $\delta_1 > \delta_2$) with a similar strong performance. This implies that this combination of β and δ values may lead to improve predictions.

Figure 4. R^2 values of the top four machine learning models across the eight cases in predicting α

Figure 5. R^2 values of the top four machine learning models across the eight cases in predicting x

Conversely, Case 3 and Case 4, where β values are not equal, but δ values are equal, make it harder to predict α. The scores of R^2 for predicting α in these cases dropped sharply, falling below 0.4 and nearing zero for Decision Tree and XGBoost. This sharp drop shows that having different β values while δ values are the same makes predicting α more difficult for the models. This contrast highlights how sensitive α is to the specific structure of interactions between β and δ.

Case 5 and Case 7, where β's are not equal and δ's are not equal, either in the same direction in Case 5 (where $\beta_1 > \beta_2$ and $\delta_1 > \delta 2$) or in opposite directions in Case 7 (where $\beta_1 > \beta_2$ and $\delta_1 < \delta_2$), showed moderate model performance for α. While predictability improved compared to Case 3 and Case 4, it did not reach the high scores observed in Case 1, Case 2, Case 8, or Case 6.

Overall, this analysis implies that while x is stable and highly predictable across all cases, α is far more sensitive to the design of the case, particularly to how β and δ interact. Therefore, selecting appropriate fuzzy probabilities combinations is essential when predicting complex targets such as α, especially in situations with stochastic or dynamic properties.

The model's ability to predict the target x changes depending on whether we compare the results across scenarios or cases. The main reason for this difference is the way the data is organized. When comparing across cases, the data for each case includes all scenarios. This means that the combinations of fuzzy probabilities remain the same, while the y-values and λ vary. In contrast, when comparing across scenarios, each scenario includes data from multiple cases. In this setup, y-values and λ are fixed, while there is more variation in the fuzzy probability values within each scenario. As a result, comparing Figure 3 with Figure 5 reveals that the model performance is more stable across cases than across scenarios. The comparison highlights how the way we organize data by cases or scenarios can significantly influence prediction accuracy. For this reason, it's crucial that decision-makers thoughtfully choose the values of β's and δ's to support better predictive outcomes than relying on scenario setup.

Besides the differences in predicting x, the contrast is even clearer when looking at how the models perform in predicting α across cases versus across scenarios. By comparing Figure 2 with Figure 4, predicting α shows less predictable performance across scenarios than across cases. Models tend to achieve consistently high R^2 scores for α, often exceeding 0.7 in many cases. The predictive performance of α varies across different cases, reflecting the influence of varying fuzzy probabilities combinations (β_1, β_2) and (δ_1, δ_2). Many cases show relatively high R^2 scores; others show drops, which give us an indication of the complexity of predicting α and its sensitivity to interactions of the fuzzy probabilities. This shows how the predictability of α is significantly shaped by the variability of fuzzy probabilities rather than by changes in the y-values of the random variable or λ.

5. Conclusions

This study presents a computational analysis grounded in simulated data derived from the Poisson distribution. The Poisson distribution is especially useful in many fields, where the frequency of arrivals, failures, or demands exhibits Poisson-like behavior but is additionally impacted by ambiguity or incomplete information. Therefore, this study gives useful insights to the decision-makers to carefully select the combinations of the required fuzzy probabilities when the Poisson distribution is used. The probabilities of the Poisson distribution were carefully generated using predetermined values of the random variable according to the corresponding mean parameter. Assumed scenarios that vary in different values of the random variable and different values of the mean parameter, as well as cases of different combinations of fuzzy probabilities, are considered.

Among different machine learning models, which were utilized, Decision Tree, XGBoost, Random Forest, and Gradient Boosting are the top four performing models across both scenarios and cases in predicting the target decision variables, x and α. However, performance differed depending on the nature of the scenarios and cases.

On the scenario level (single scenario for all cases), the models predict α better than x. The scenarios showed that as the values of the random variable became more dispersed about the same value of the mean parameter, the prediction of α is getting better. In general, α is easier for the models to predict than x across different scenarios, but its predictability still depends on how the dispersion of values aligns with the mean parameter value, which plays a key role in the accuracy of the prediction. On the case level (single case for all scenarios), the results showed that while x remains stable and highly predictable across all fuzzy probabilities, α is far more sensitive to the design of the case. For this reason, it's crucial that decision-makers thoughtfully choose the values of fuzzy probabilities to support better predictive outcomes than relying on scenario-based data. Additionally, this demonstrates how the predictability of α is significantly shaped by the variability of fuzzy probabilities rather than by changes in values of the random variable or the mean parameter value.

Therefore, this paper provides the decision-maker with a machine learning decision support tool, especially when there is a lack of information about the probabilities of the values of the used discrete random variable, as well as when the probabilities that are required to be achieved are imprecise.

Finally, the approach of this study can be utilized for other discrete probability distributions, and for different linear combinations of x's instead of just x. This may be considered in future studies.

REFERENCES

1. Abdelaziz, F. B. and Masri, H. (2005). Stochastic programming with fuzzy linear partial information on probability distribution. *European Journal of Operational Research*, **162**(3), 619–629.
2. Aiche, F., Abbas, M. and Dubois, D. (2013). Chance-constrained programming with fuzzy stochastic coefficients. *Fuzzy Optimization and Decision Making*, **12**, 125–152.
3. Alqasem, O. A., Abd Elwahab, M. E., Bakr, M., Al-Sharari, H. D. and Elsharkawy, K. (2024). Artificial neural network method in solving smart cities construction fuzzy multi-objective linear programming problems. *AIP Advances*, **14**(7).
4. Bahad, P. and Saxena, P. (2020). Study of adaboost and gradient boosting algorithms for predictive analytics. In *Proceedings of ICSC 2019: International Conference on Intelligent Computing and Smart Communication*, Springer Singapore, 235–244.
5. Bakhtavar, E., Sadiq, R. and Hewage, K. (2021). Optimization of blasting-associated costs in surface mines using risk-based probabilistic integer programming and firefly algorithm. *Natural Resources Research*, **30**(6), 4789–4806.
6. Bhowmick, A., Chakraverty, S. and Chatterjee, S. (2024). Parametric Optimization for Fully Fuzzy Linear Programming Problems with Triangular Fuzzy Numbers. *Mathematics*, **12**(19), 3051.
7. Chen, S. P. (2007). Solving fuzzy queueing decision problems via a parametric mixed integer nonlinear programming method. *European Journal of Operational Research*, **177**(1), 445–457.
8. Divya, P. (2012). Quality interval acceptance single sampling plan with fuzzy parameter using poisson distribution. *International Journal of Advanced Research in Technology*, **1**(3), 115–125.
9. Effati, S., Pakdaman, M. and Ranjbar, M. (2011). A new fuzzy neural network model for solving fuzzy linear programming problems and its applications. *Neural Computing and Applications*, **20**, 1285–1294.
10. Figueroa–García, J. C., Hernández, G. and Franco, C. (2022). A review on history, trends and perspectives of fuzzy linear programming. *Operations Research Perspectives*, **9**, 100247.
11. Ghanbari, R., Ghorbani-Moghadam, K., Mahdavi-Amiri, N. and De Baets, B. (2020). Fuzzy linear programming problems: models and solutions. *Soft Computing,* **24**(13), 10043–10073.
12. Iskander, M. G. (2012). An approach for linear programming under randomness and fuzziness: A case of discrete random variables with fuzzy probabilities. *International Journal of Operational Research,* **15**(2), 215–225.
13. Iskander, M. G. (2014). A suggested approach for solving fuzzy stochastic multiobjective programming problems in the case of discrete random variables. *International Journal of Mathematics in Operational Research,* **6**(2), 258–269.
14. Ji, L., Niu, D. X., Xu, M. and Huang, G. H. (2015). An optimization model for regional micro-grid system management based on hybrid inexact stochastic-fuzzy chance-constrained programming. *International Journal of Electrical Power& Energy Systems,* **64**, 1025–1039.
15. Jiménez, M., Arenas, M., Bilbao, A. and Rodríguez, M. V. (2007). Linear programming with fuzzy parameters: An interactive method resolution. *European Journal of Operational Research,* **177**(3), 1599–1609.
16. Kalaichelvan, K., Ramalingam, S., Dhandapani, P. B., Leiva, V. and Castro, C. (2024). Optimizing the economic order quantity using fuzzy theory and machine learning applied to a pharmaceutical framework. *Mathematics,* **12**(6), 819.
17. Kumar, A. and Mishra, B. (2024). Nonlinear fuzzy chance constrained approach for multi-objective mixed fuzzy-stochastic optimization problem. *Opsearch,* **61**(1), 121–136.
18. Kumar, A., Kaur, J. and Singh, P. (2011). A new method for solving fully fuzzy linear programming problems. *Applied Mathematical Modelling,* **35**(2), 817–823.
19. Li, S. (2010). Poisson process with fuzzy rates. *Fuzzy Optimization and Decision Making,* **9**, 289–305.
20. Li, Y., Lin, F., Yang, Z., Fang, X. and Lu, X. (2025). Energy management approach for wayside energy storage system in urban rail transit considering real-observable characteristics: A deep reinforcement learning method based on fuzzy logic guided. *Journal of Energy Storage,* **122**, 116676.
21. Liu, B. (2001). Fuzzy random chance-constrained programming. *IEEE Transactions on Fuzzy Systems,* **9**(5), 713–720.
22. Luhandjula, M. K. (2006). Fuzzy stochastic linear programming: survey and future research directions. *European Journal of Operational Research,* **174**(3), 1353–1367.
23. Maulud, D. and Abdulazeez, A. M. (2020). A review on linear regression comprehensive in machine learning. *Journal of Applied Science and Technology Trends,* **1**(2), 140–147.
24. Mienye, I. D., Sun, Y. and Wang, Z. (2019). Prediction performance of improved decision tree-based algorithms: a review. *Procedia Manufacturing,* **35**, 698–703.
25. Najm, A. A. and Ali, B. K. (2024). Modeling Biological Uncertainty Using the Double Fuzzy Poisson Distribution. *Letters in Biomathematics,* **1**(1), 51–60.
26. Nguyen, V. H. (2007). Solving linear programming problems under fuzziness and randomness environment using attainment values. *Information Sciences,* **177**(14), 2971–2984.
27. Quan, J., Peng, Y., and Su, L. (2025). Logistics demand prediction using fuzzy support vector regression machine based on Adam optimization. *Humanities and Social Sciences Communications,* **12**(1), 1–13.
28. Rajan, M. P. (2022). An efficient Ridge regression algorithm with parameter estimation for data analysis in machine learning. *SN Computer Science,* **3**(2), 171.
29. Ramraj, S., Uzir, N., Sunil, R. and Banerjee, S. (2016). Experimenting XGBoost algorithm for prediction and classification of different datasets. *International Journal of Control Theory and Applications,* **9**(40), 651–662.
30. Salman, H. A., Kalakech, A. and Steiti, A. (2024). Random forest algorithm overview. *Babylonian Journal of Machine Learning,*, 69–79.
31. Tanaka, H., Guo, P. and Türksen, I. B. (2000). Portfolio selection based on fuzzy probabilities and possibility distributions. *Fuzzy Sets and Systems,* **111**(3), 387–397.
32. Yano, H. (2020). Multiobjective two-level simple recourse programming problems with discrete-type LR fuzzy random variables. *Procedia Computer Science,* **176**, 531–540.

33. Yuvashri, P. and Saraswathi, A. (2024). A novel approach for multi-objective linear programming model under spherical fuzzy environment and its application. *Journal of Intelligent & Fuzzy Systems,* **46**(2), 3259–3280.
34. Zahiri, B., Torabi, S. A. and Tavakkoli-Moghaddam, R. (2017). A novel multi-stage possibilistic stochastic programming approach (with an application in relief distribution planning). *Information Sciences,* **385**, 225–249.
35. Zhang, Z. (2016). Introduction to machine learning: k-nearest neighbors. *Annals of Translational Medicine,* **4**(11), 218.
36. Zimmermann, H.-J. (1978). Fuzzy programming and linear programming with several objective functions. *Fuzzy Sets and Systems,* **1**(1), 45–55.