Original Research Article

Use of one-factor design of experiments (DOE) for Regression Modelling and Validation: A Robust Methodology

Running Title: One-factor design of experiments (DOE) for Regression Modelling

Abstract:

In the present research era, high accuracy methods as a statistical analysis tool are increasing. Therefore, researchers are more focused to produce reliable and accurate results. Hence, the use of data modelling techniques is more focused to meet the needs of the current research trend. On the other hand, Design of Experiment (DOE) is extensively used among various scientific fields; however, its limitations do not allow these study designs for modelling purposes. Therefore, this study was designed to develop a methodology combining statistical methods that can provide to use one-factor DOE study designs for modelling and predictions. The addition of Fuzzy regression and multilayer feedforward (MLFF) neural network along with multiple linear regression would provide more accurate results with high accuracy. Furthermore, the developed methodology was tested on a dataset to test the methodology's performance and results provided that methodology provided regression models through MLR and fuzzy with high accuracy with the testing of the model's predictability through MLFF.

Keywords:
Design of experiment, Regression, Methodology, Robust, MLFF

Introduction:

DOE study designs are widely used among various scientific and non-scientific experiments and can be used to explore or study the relationship or association among the variables [1-3]. However, the use of DOE study designs is limited to studying the association between the variables and cannot be used for prediction purposes [4]. On the other hand, forecasting is becoming popular and shared in the studies as it helps to improve the significance of the study findings and impact of the research; hence the use of regression modelling has become increasing among the scientific community.

Due to the DOE limitations, these study designs cannot be used for regression modelling. Hence, some prior work, which is called data transformation, is required to use DOE study designs for regression modelling [5]. Furthermore, due to the nature of the independent variables in the DOE study designs, called factors, it is likely to have fuzziness in the transformed data. On the other hand, linear regression models are designed to model crisp datasets, and their aptness becomes poor in case of vague data [6]. Therefore, to address the issue of data fuzziness and incorporate fuzzy data into the regression model, a new approach in regression modelling was introduced, which is called Fuzzy linear regression modelling [7]. Like linear regression, which is based on probability theory, fuzzy regression is based on the theory of possibility [8,9]. L.A. Zadeh did the initial work on fuzzy regression, which later proceeded by Tanaka, Diamond, Ishibuchi and others [8,10,11].

Therefore, this study was designed to provide a methodology that can enable researchers to use their DOE study designs for prediction purposes. The study aim was to provide a comprehensive and robust methodology that included the transformation of one-factor DOE study design to linear form, use of bootstrapping to enhance the accuracy of estimated regression parameters, use of linear and fuzzy regression models and utilization of multilayer feedforwarding (MLFF) neural networking for model validation.

Methodology:

One factor DOE study design was used for transformation into linear form, followed by regression modelling and validation. Therefore, the data transformation process on generalized one-factor DOE study design was initially elaborated. Hence, Table 1 introduced the distribution of one-factor DOE with *i* treatments and *j* observations within each treatment.

Table 1: General data distribution for one-factor study design

	Treatment		
1	**2**	**…**	**i**
y_{11}	y_{21}	…	y_{i1}
y_{12}	y_{22}	…	y_{i2}
y_{13}	y_{23}	…	y_{i3}
.	.	…	.
.	.		.
.	.		.
y_{1j}	y_{2j}	…	y_{ij}

Data presented in table 3.1 has one dependent variable (y) and "*i*" treatments (factors). However, to transform the data into the linear form, "i^{th}" treatment or factor is not required, as expressed in terms of i-1 treatments or factors [12]. Therefore, a generalized matrix for the transformed dataset contained $r=i$-1 and $n=j$. Now the utmost part of the transformation process is to code the factor. The variables generated after the coding are called indicator variables that take on values 0, 1 or -1 [5]. This coding process must be done carefully because it leads to the regression coefficients in the β vector. Hence, the matrix obtained after the transformation of the dataset has dependent and independent variables matrix, matrix for slopes and matrix for random error.

$$Y = \begin{bmatrix} Y_{11} \\ Y_{12} \\ \vdots \\ Y_{1n} \\ Y_{21} \\ Y_{22} \\ \vdots \\ Y_{in} \end{bmatrix}; X = \begin{bmatrix} 1 & 1 & 0 & \cdot & 0 \\ 1 & 1 & 0 & \cdot & 0 \\ 1 & 1 & 0 & \cdot & 0 \\ 1 & 1 & 0 & \cdot & 0 \\ 1 & 1 & 0 & \cdot & 0 \\ 1 & 0 & 1 & \cdot & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & -1 & -1 & \cdot & -1 \end{bmatrix}; \beta = \begin{bmatrix} \mu. \equiv \beta_0 \\ \tau_1 = \beta_1 \\ \tau_2 = \beta_2 \\ \tau_3 = \beta_3 \\ \vdots \\ \tau_{i-1} = \beta_{i-1} \end{bmatrix}; \begin{bmatrix} \varepsilon_{11} \\ \varepsilon_{12} \\ \vdots \\ \varepsilon_{1n} \\ \varepsilon_{21} \\ \varepsilon_{22} \\ \vdots \\ \varepsilon_{in} \end{bmatrix}$$

Let X_{ij1} denote the value of indicator variable X_1, X_{ij2} indicate the value of indicator variable X_2, and so on. Using i-1 indicators in the model and multiple linear regression model can be stated as

$$Y_{ij} = \mu. + \tau_1 x_{ij1} + \tau_2 x_{ij2} + \cdots + \tau_{i-1} x_{ij(i-1)} + \varepsilon_{ij}$$

where,

$$x_{ij1} = \begin{matrix} 1 & \textit{if the case from the factor level} 1 \\ -1 & \textit{if the case from the factor level} (i) \\ 0 & \textit{Otherwise} \end{matrix}$$

$$x_{ij2} = \begin{matrix} 1 & \textit{if the case from the factor level} 2 \\ -1 & \textit{if the case from the factor level} (i) \\ 0 & \textit{Otherwise} \end{matrix}$$

$$x_{ij(i-1)} = \begin{matrix} 1 & \text{if the case from the factor level } (\mathrm{i}-1) \\ -1 & \text{if the case from the factor level } (i) \\ 0 & \text{Otherwise} \end{matrix}$$

Therefore, to apply the above process of data transformation to get the linear form, a one-factor DOE data was extracted from a book "Probability & Statistics for Engineers & Scientists" by Walpole R.E et al. The dataset consisted of 25 patients with a fever of 38 degrees Celsius or higher and used five different brands of headache relief medications. The number of hours of headache relief was recorded in Table 2.

Table 2: Hours of relief from five different brands of headache tablets

Group 1	Group 2	Group 3	Group 4	Group 5
5.2	9.1	3.2	2.4	7.1
4.7	7.1	5.8	3.4	6.6
8.1	8.2	2.2	4.1	9.3
6.2	6.0	3.1	1.0	4.2
3.0	9.1	7.2	4.0	7.6

After that, the transformation process was initiated; data in Table 2 had five groups (r = 5) and five observations in each group (n = 5). Hence, each column of the transformed matrix had 25 observations ($r \times n$). The independent variables ' matrix required indicator variables with values ranging from 0, 1 and -1. Therefore, the matrix after transformation looked like this:

$$Y = \begin{bmatrix} Y_{11} \\ Y_{12} \\ Y_{13} \\ Y_{14} \\ Y_{15} \\ Y_{21} \\ \vdots \\ Y_{55} \end{bmatrix}; X = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & -1 & -1 & -1 & -1 \end{bmatrix}; \beta = \begin{bmatrix} \mu. \equiv \beta_0 \\ \tau_1 = \beta_1 \\ \tau_2 = \beta_2 \\ \tau_3 = \beta_3 \\ \tau_4 = \beta_4 \end{bmatrix}; \begin{bmatrix} \varepsilon_{11} \\ \varepsilon_{12} \\ \varepsilon_{13} \\ \varepsilon_{14} \\ \varepsilon_{15} \\ \varepsilon_{21} \\ \vdots \\ \varepsilon_{55} \end{bmatrix}$$

Let X_{i1} denoted the value of indicator variable X_1, X_{i2} indicated the value of indicator variable X_2, etc. Using t-1 indicators in the model and multiple linear regression model for the study would be stated as

$$Y_i = \mu. + \tau_1 x_{i1} + \tau_2 x_{i2} + \tau_3 x_{i3} + \tau_4 x_{i4} + \varepsilon_i$$

where

$$x_{i1} = \begin{matrix} 1 & \text{if the case from the factor level 1} \\ -1 & \text{if the case from the factor level 5} \\ 0 & \text{Otherwise} \end{matrix} \qquad x_{i2} = \begin{matrix} 1 & \text{if the case from the factor level 2} \\ -1 & \text{if the case from the factor level 5} \\ 0 & \text{Otherwise} \end{matrix}$$

$$x_{i3} = \begin{matrix} 1 & \text{if the case from the factor level 3} \\ -1 & \text{if the case from the factor level 5} \\ 0 & \text{Otherwise} \end{matrix} \qquad x_{i4} = \begin{matrix} 1 & \text{if the case from the factor level 4} \\ -1 & \text{if the case from the factor level 5} \\ 0 & \text{Otherwise} \end{matrix}$$

Table 3 represents the data after transformation. The data had one dependent variable Y_{ij}, and four X_{i1}, X_{i2}, X_{i3}, and X_{i4}. Data now transformed to linear form and could be used for regression modelling.

Table 3: Regression approach to one-factor DOE

Y_{ij}	X_{i1}	X_{i2}	X_{i3}	X_{i4}
5.2	1	0	0	0
4.7	1	0	0	0
8.1	1	0	0	0
6.2	1	0	0	0
3.0	1	0	0	0
⋮	⋮	⋮	⋮	⋮
4.2	-1	-1	-1	-1
7.6	-1	-1	-1	-1

Hence, data tabulated in table 3 showed a transformed form of one-factor DOE into linear. Therefore, this can be used for regression modelling. Thus, the R-software was used for writing syntax for regression modelling and validation. In this developed methodology, to enhance the accuracy of estimated regression parameters, the syntax for bootstrapping was utilized first after the data entry into the R. syntax for data splitting was followed by bootstrapping, which provided an opportunity to have another (independent) dataset that could be used to test the developed model. Hence, the syntax for data splitting was used to get "train" and test" datasets in 70:30 ratio. After that, the syntax for multiple linear regression and fuzzy regression was used by using "train" data, and the mean square error for the model was calculated by using "test" data. To further validate the model and to check how far the model's prediction is from reality, the syntax for multilayer feedforward neural network (MLFF) was used. Since normalization is necessary before performing neural networks, the syntax for data normalization utilized prior neural networks [13]. Furthermore, normalized data were split into "train" and "test", in which "train" data was used for building the architecture of the neural network and "test" data used to test the predictability of the developed network. Therefore, comprehensive, combine and robust methodology with R-syntax by using the data in table 3 is as follows:

```
y = c(5.2, 4.7, 8.1, 6.2, 3.0, 9.1, 7.1, 8.2, 6.0, 9.1, 3.2, 5.8, 2.2, 3.1,
      7.2, 2.4, 3.4, 4.1, 1.0, 4.0, 7.1, 6.6, 9.3, 4.2, 7.6)
x1 = c(1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1,
       -1, -1, -1, -1)
x2 = c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1,
       -1, -1, -1, -1)
x3 = c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, -1,
       -1, -1, -1, -1)
x4 = c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, -1,
```

```
    -1, -1, -1, -1)

X = cbind(x1,x2,x3,x4)
data = data.frame(y,x1,x2,x3,x4)
```

R Syntax for data entry and Bootstrapping

```
#/Performing Bootstrap for 1000

mydata <- rbind.data.frame(data, stringsAsFactors = FALSE)
iboot <- sample(1:nrow(mydata),size=1000, replace = TRUE)
bootdata <- mydata[iboot,]
```

R Syntax for splitting booted data into test and train data

```
#/Randomly split the data into 70:30
#70 percent of the data at our disposal to train the network
#30 percent to test the network/

smp_size <- floor(0.70*nrow(bootdata))
set.seed(123)
train_ind<- sample(seq_len(nrow(bootdata)), size=smp_size)

train <- data.frame(bootdata[train_ind,])
test <- data.frame(bootdata[-train_ind,])

# Print Data
print(train)
print(test)
```

R Syntax for MLR Regression Modelling

```
# /Fit a Linear Regression Model
# Use Mean Squared Error (MSE) as a Measure of Prediction Performance/
#/Predict the Values for the Test Set and Calculate the MSE/
Model <- lm(y~x1+x2+x3+x4, data=train)
summary(Model)
predict_lm <- predict(Model,test)
MSE.lm <- sum((predict_lm - test$y)^2)/nrow(test)
```

```
MSE.lm
```

Syntax for Fuzzy regression Modelling

```
if(!require(fuzzyreg)) install.packages("fuzzyreg", dependencies = TRUE)
library(fuzzyreg)
```

##Fuzzy linear model using the PLRLS method##

```
f <-fuzzylm(y ~ x1+x2+x3+x4, data=train$lee, method = "plrls", fuzzy.left.x = NULL, fuzzy.right.x = NULL, fuzzy.left.y = NULL, fuzzy.right.y = NULL)
coef(f)
```

R Syntax for data normalization and Multilayer Feedforward Neural Network

#/Performing neural network

#/install the neuralnet package/

```
if(!require(neuralnet)){install.packages("neuralnet")}
library("neuralnet")
```

#/Scaling the data for normalization

 # Method (usually called feature scaling) to get all the scaled data

 # in the range [0,1]/

```
max_data <- apply(bootdata, 2, max)
min_data <- apply(bootdata, 2, min)
data_scaled <- scale(bootdata,center = min_data, scale = max_data - min_data)
```

#/Randomly split the data into 70:30

#70 percent of the data at our disposal to train the network

#30 percent to test the network/

```
index = sample(1:nrow(bootdata),round(0.70*nrow(bootdata)))
train_data <- as.data.frame(data_scaled[-index,])
test_data <- as.data.frame(data_scaled[-index,])
```

#/Build the network

#Create 2 hidden layers have 3 and 2 neurons respectfully

#Input layer = 4

#Output layer = 1/

```
n = names(bootdata)
f = as.formula(paste("y ~", paste(n[!n %in% "y"], collapse = " + ")))
nn = neuralnet(f,data=train_data,hidden=c(3,2),linear.output=T)
plot(nn)
```

```
options(warn=-1)

#/30 percent of the available data to do this:
#using only the first 2 columns representing the input variables
#of the network and 1 is the output for NN/
predicted <- compute(nn,test_data[,1:4])

#/Use the Mean Squared Error NN (MSE-forecasts the network) as a measure
of how far away our predictions are from the real data/
MSE.net <- sum((test_data$y - predicted$net.result)^2)/nrow(test_data)
MSE.net

#/Printing the Value of MSE for Linear Model and Neural Network/
print(paste(MSE.lm,MSE.net))
```

Results:

The outcome generated from data after running the R syntax were summarized in this section. Parameters for multiple linear regression were calculated first with the summary of the model. Table 4 summarizes the output for the MLR estimated parameters. Slopes for the parameters $x2$, $x3$ and $x4$ were statistically significant (p <0.001).

Table 4: Parameter Estimates of Regression Modelling

Parameter Estimates				
Variable	Parameter Estimate	Standard Error	t-value	P-value
Intercept	5.56	0.15	35.87	<.0001
x1	-0.104	0.315	-0.329	0.235
x2	2.55	0.306	8.304	<.0001
x3	-2.43	0.325	-7.473	<.0001
x4	-2.03	0.325	-6.244	<.0001

Hence, a multiple linear regression model with estimated parameters can be written as

Hours of relief = 5.56 - 0.1.04 x1 + 2.55 x2 - 2.43 x3 - 2.03 x4

The model was statistically significant (P<0.0001), and the adjusted R-square was 0.71 (Table 5). To determine the predictability of the MLR model, the MSE (mean square error) of the model was computed, and it was found to be 1.05 (Table 5).

Table 5: MLR Model Summary

Residual SE	1.279	R-Square	0.727
MSE	1.05	Adj R-Sq	0.71
F-statistic	43.36	P-value	<0.0001

Similarly, table 6 tabulated the parameters obtained for fuzzy regression, containing Central, lower and upper boundary values for intercept and variables.

Table 6: Parameter Estimates of Fuzzy Modelling

	Parameter Estimates		
Variable	Central Tendency	Lower Boundary	Upper Boundary
Intercept	5.51	3.68	6.78
*x*1	-0.007	-0.68	1.32
*x*2	2.32	2.32	2.32
*x*3	-1.15	-1.48	0.42
*x*4	-2.68	-2.68	-2.68

To draw the fuzzy regression equation, a central tendency column was used. However, lower and upper boundary columns were used from table 6 to construct the equation for lower and upper boundaries for boundaries of fuzzy regression. Therefore, the fuzzy regression equations are as follow :

Central tendency of the fuzzy regression model:

Hours of relief = 5.51 - 0.007x1 + 2.32x2 - 1.15x3 – 2.68x4

Lower boundary of the model support interval:

Hours of relief = 3.68 - 0.6845x1 + 2.32x2 - 1.48x3 – 2.68x4

Upper boundary of the model support interval:

Hours of relief = 6.78 + 1.32x1 + 2.32x2 + 0.42x3 – 2.68x4

To quantify that how close the predicted values of the dependent variable (Y) through each model (MLR and Fuzzy), the equations derived above, from MLR and fuzzy, were used on test data to calculate values for predicted Y (Table 7). Furthermore, the absolute difference between original and predicted Y from each model was computed to calculate the numeric difference between original and predicted values of the dependent variable through each model. Therefore, the mean of the absolute difference between MLR and fuzzy was very small; however, fuzzy was marginally better than MLR.

Table 7: Predicted values of Y form MLR and fuzzy models

	Predicted Y		Abs difference	
Y	MLR	Fuzzy	MLR	Fuzzy
9.1	7.5791	7.027	1.5209	2.073
4.2	3.128	4.36	1.072	0.16
6.6	3.528	2.83	3.072	3.77
7.1	7.5791	7.027	0.4791	0.073
6.6	3.528	2.83	3.072	3.77
.	.	.	.	.
.	.	.	.	.
5.8	8.1089	7.83	2.3089	2.03
9.1	8.1089	7.83	0.9911	1.27
8.1	7.5791	7.027	0.5209	1.073
Average			**2.16**	**2.01**

MLFF neural network was embedded in the syntax to test the strength of the parameters used in the regression model; therefore, it helped determine how good the forecasting was through the derived regression model. Figure 1 presents the architecture of the neural network obtained from the data. The figure had hidden layers with 3 and 2 neurons, respectively, and as the networking was Feedforward, the information was only carried in the forward direction (figure 1). Furthermore, the calculated MSE for the network was 0.09. Therefore, the minimal error measurement demonstrated that constructed model's accuracy and forecasting capability. As a result, all four independent variables were good predictors of hours of headache relief after taking medicine.

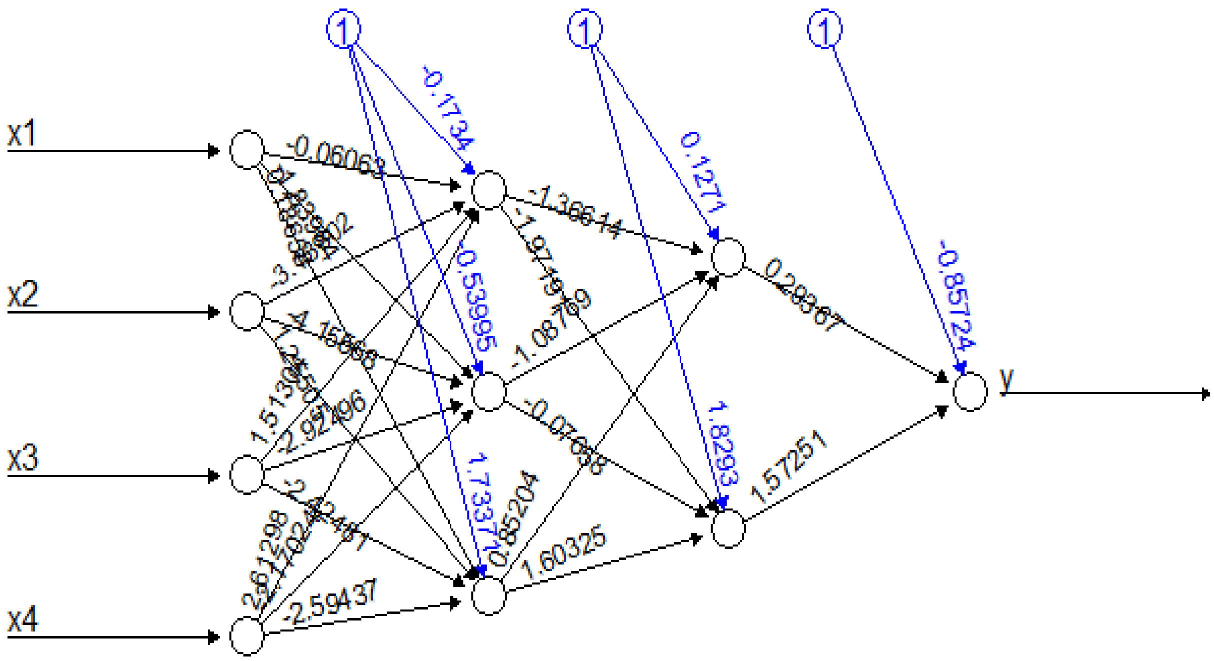

Figure 1: The architecture of the MLFF neural network with four input variables, two hidden layers and one output node

Conclusion:

This study provided a combined, comprehensive and robust methodology for using one-factor DOE for prediction purposes by transforming DOE into a linear form. Using an independent dataset "test data" to find the predictability of MLR by calculating MSE provided the precision and predictability of the derived model. In addition, this methodology also embedded the fuzzy regression to enhance the predictability and accuracy of the model in case of fuzziness in the data and model performance was evaluated (Table 7) and found estimation from fuzzy was comparatively better. Furthermore, small MSE from the neural network provided that predictions or estimates drawn from the obtained regression model would be very close to actual population characteristics. Therefore, this methodology can allow the research community and academicians to use their DOE study designs for prediction purposes to meet the growing needs of research trends and help them get more improved outcomes from their research.

References:

1. Gu, S., Wang, X., Shi, L., Sun, Q., Hu, X., Gu, Y., Sun, X. & Dong, H. (2020) Health-related quality of life of type 2 diabetes patients hospitalized for a diabetes-related complication. Quality of Life Research, 29(10), 2695-704.

2. Memon, J., Holakouie-Naieni, K., Majdzadeh, R., Yekaninejad, M.S., Garmaroudi, G., Raza, O. & Nematollahi, S. (2019) Knowledge, attitude, and practice among mothers about newborn care in Sindh, Pakistan. BMC pregnancy and childbirth, 19(1), 1-9.

3. Hamid, S.K., Al-Dubayan, A.H., Al-Awami, H., Khan, S.Q. & Gad, M.M. (2019) In vitro assessment of the antifungal effects of neem powder added to polymethyl methacrylate denture base material. Journal of clinical and experimental dentistry, 11(2), e170.

4. Bas, E., Yolcu, U., & Egrioglu, E. (2021). Intuitionistic fuzzy time series functions approach for time series forecasting. Granular Computing, 6(3), 619-629.

5. Khan, S.Q., Ahmed, W.M.A.W., Aleng, N.A. (2121) The development of regression modeling through design of experiment (doe) methodology: an overview the effect of bootstrapping data. Sapporo Medical Journal, 55(10), 2010215510396.https://www.maejournal.com/article/the-development-of-regression-modeling-through-design-of-experiment-doe-methodology-an-overview-the-effect-of-bootstrapping-data

6. Van-Kuijk, S.M., Dankers, F.J., Traverso, A. & Wee, L. (2019) Preparing data for predictive modelling. Fundamentals of clinical data science, 75-84.

7. Hesamian, G., & Akbari, M. G. (2020). A fuzzy additive regression model with exact predictors and fuzzy responses. Applied Soft Computing, 95, 106507.

8. Klir, G., & Yuan, B. (2020). Possibility theory. In Handbook of Fuzzy Computation (pp. 185-B4). CRC Press.

9. Feng, X., Tan, Q. & Wei, C. (2018) Hesitant fuzzy linguistic multi-criteria decision making based on possibility theory. International Journal of Machine Learning and Cybernetics, 9(9), 1505-17.

10. Škrjanc, I., Iglesias, J.A., Sanchis, A., Leite, D., Lughofer, E. & Gomide, F. (2019) Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification: a survey. Information Sciences, 490, 344-68.

11. López, S., Márquez, A.A., Márquez, F.A. & Peregrín, A. (2019) Evolutionary design of linguistic fuzzy regression systems with adaptive defuzzification in big data environments. Cognitive Computation, 11(3), 388-99.

12. Ahmed, W.M.A.W., Khan, S.Q., Rohim, R.A.A., Aleng, N.A., Ghazali, F.M.M. (2020) Approximation of Randomized Block Design Towards Fuzzy Multiple Linear Regression: A Case Study In Health Sciences. International Journal of Scientific and Technology Research, 9(1): 1303-8.

13. Singh, D., & Singh, B. (2020). Investigating the impact of data normalization on classification performance. Applied Soft Computing, 97, 105524.

Table 1: General data distribution for one-factor study design

Treatment			
1	**2**	**…**	**i**
y_{11}	y_{21}	…	y_{i1}
y_{12}	y_{22}	…	y_{i2}
y_{13}	y_{23}	…	y_{i3}
.	.	…	.
.	.		.
.	.		.
y_{1j}	y_{2j}	…	y_{ij}

Table 2: Hours of relief from five different brands of headache tablets

Group 1	Group 2	Group 3	Group 4	Group 5
5.2	9.1	3.2	2.4	7.1
4.7	7.1	5.8	3.4	6.6
8.1	8.2	2.2	4.1	9.3
6.2	6.0	3.1	1.0	4.2
3.0	9.1	7.2	4.0	7.6

Table 3: Regression approach to one-factor DOE

Y_{ij}	X_{i1}	X_{i2}	X_{i3}	X_{i4}
5.2	1	0	0	0
4.7	1	0	0	0
8.1	1	0	0	0
6.2	1	0	0	0
3.0	1	0	0	0
⋮	⋮	⋮	⋮	⋮
4.2	-1	-1	-1	-1
7.6	-1	-1	-1	-1

Table 4: Parameter Estimates of Regression Modelling

	Parameter Estimates			
Variable	Parameter Estimate	Standard Error	t-value	P-value
Intercept	5.56	0.15	35.87	<.0001
x1	-0.104	0.315	-0.329	0.235
x2	2.55	0.306	8.304	<.0001
x3	-2.43	0.325	-7.473	<.0001
x4	-2.03	0.325	-6.244	<.0001

Table 5: MLR Model Summary

Residual SE	1.279	R-Square	0.727
MSE	1.05	Adj R-Sq	0.71
F-statistic	43.36	P-value	<0.0001

Table 6: Parameter Estimates of Fuzzy Modelling

	Parameter Estimates		
Variable	Central Tendency	Lower Boundary	Upper Boundary
Intercept	5.51	3.68	6.78
$x1$	-0.007	-0.68	1.32
$x2$	2.32	2.32	2.32
$x3$	-1.15	-1.48	0.42
$x4$	-2.68	-2.68	-2.68

Table 7: Predicted values of Y form MLR and fuzzy models

	Predicted Y		Abs difference	
Y	MLR	Fuzzy	MLR	Fuzzy
9.1	7.5791	7.027	1.5209	2.073
4.2	3.128	4.36	1.072	0.16
6.6	3.528	2.83	3.072	3.77
7.1	7.5791	7.027	0.4791	0.073
6.6	3.528	2.83	3.072	3.77
.	.	.	.	.

.	.	.	.	.
5.8	8.1089	7.83	2.3089	2.03
9.1	8.1089	7.83	0.9911	1.27
8.1	7.5791	7.027	0.5209	1.073
Average			**2.16**	**2.01**

Figure 1: The architecture of the MLFF neural network with four input variables, two hidden layers and one output node