International Journal of Statistical Sciences *ISSN 1683-5603*
Vol. 21(1), 2021, pp 127-146

Landslide Susceptibility Zonation, Using Machine Learning Techniques, in a Part of the Hilly Areas of the Darjeeling District, West Bengal

Diptarshi Mitra[1] and Asim Ratan Ghosh[2]*

[1]Computer Science and Engineering Wing, Directorate of Distance Education, Annamalai University, Annamalainagar-608002, India; current (residential) address: BB-35/5, Salt Lake City, Kolkata-700064, India

[2]Department of Science and Technology and Biotechnology, Government of West Bengal, Bikash Bhawan (4th Floor), Salt Lake City, Kolkata-700091, India

*Correspondence should be addressed to Asim Ratan Ghosh (Email: asimrghosh@gmail.com)

[Received November 10, 2020; Revised January 5, 2021; Accepted February 20, 2021]

Abstract

This study attempts Landslide Susceptibility Zonation of a part of the Kurseong Subdivision of the Darjeeling District of West Bengal, using Machine Learning techniques. Here, two models of Machine Learning, viz., Logistic Regression, implemented using Python programming, and Artificial Neural Network, implemented using Python and MATLAB programming, have been employed. And, four causative factors viz., land use/land cover, lithostratigraphy, structural features, and slope angle, have been considered, for susceptibility zonation. Also, data regarding the past landslide events in the study area, which occurred in or before 2015, have been used in this work. A part of the whole dataset pertaining to the causative factors and past landslide events, has been utilized for training and testing the models. While testing, the models exhibit high levels of accuracy. Then the models are applied to the whole dataset, and the output is recorded. For Artificial Neural Network model, implemented with MATLAB, the output is adjusted to keep it within 0 and 1. The output/adjusted output of the models indicates the probability of being susceptible to landslide/s. Finally, this output/adjusted output is classified and converted into Landslide Susceptibility Zonation maps. The zones of highest susceptibility, in these three Landslide Susceptibility Zonation maps, generated by Logistic Regression and Artificial Neural Network models, are mostly located in the central, northern, and northeastern parts of the study area; these zones reasonably agree with the sites of the previous landslide occurrences.

Keywords: Landslide Susceptibility Zonation, Kurseong Subdivision, Logistic Regression, Artificial Neural Network, Python, MATLAB.

AMS Classification: 68T99.

1. Introduction

Landslides are natural hazards, common in the hilly regions, which often result in a considerable number of human deaths and injuries, and a great loss of property, infrastructure and natural resources (Marrapu and Jakka, 2014; Pourghasemi, Gayen, Park, Lee, and Lee, 2018; Zhou et al., 2018).

A number of factors, e.g., geomorphology, structural features, anthropogenic activities, climate etc., are instrumental in the occurrences of landslides (Kanungo, Arora, Sarkar, and Gupta, 2009). As landslides are difficult to predict, Landslide Susceptibility Zonation (LSZ), which categorizes regions into various classes according to their vulnerabilities to the occurrences of landslides, is extremely important for planning developmental activities, and reducing the loss of life and resources, in the hilly regions (Chauhan, Sharma, Arora, and Gupta, 2010; Kadavi, Lee, and Lee, 2018).

This study attempts to create Landslide Susceptibility Zonation map of a part of the hilly regions of the Darjeeling District, in the state of West Bengal, in India; Darjeeling Hills constitute a portion of the Eastern Himalayas (Chawla et al., 2018). To be more precise, the study area for this work is a part of the Kurseong Subdivision of the Darjeeling District; this region is highly susceptible to landslides (Chawla et al., 2018).

For understanding landslide susceptibility, this study has used four important causative factors viz., land use/land cover, lithostratigraphy, structural features and slope angle (in degree). Land use shows how people utilize the landscape (e.g., by building residential areas, creating agricultural lands etc.) (National Oceanic and Atmospheric Administration (NOAA), 2018); land cover depicts the natural features (e.g., forests, wetlands etc.) which cover the region (National Oceanic and Atmospheric Administration (NOAA), 2018); lithostratigraphy deals with the lithologic properties and their stratigraphic relations; structural features refer to the geologic structures (faults, thrusts, joints, folds etc.) generated due to the powerful tectonic forces that occur within the earth; slope angle denotes the angle of the slope of the terrain. And Machine Learning techniques have been utilized to generate the LSZ maps, using these aforesaid factors as input.

Machine Learning is the science and art of programming computers to enable them to learn from data (Géron, 2017). In other words, one may consider that a program has learnt from data, with regard to some task, and some performance measure, if the performance of the program relating to the task, as indicated by the performance measure, improves after the program is fed with the data (Géron, 2017). The process of feeding data to the program, so that it can improve its

performance, is called training, and the data used for training, is called training data (Géron, 2017). Programs involving Machine Learning are most likely to be more accurate than the traditional programs (Géron, 2017). In this work, two important Machine Learning models have been employed, viz., Logistic Regression (LR) and Artificial Neural Network (ANN); both of these models have been used by quite a number of researchers working on landslides (Chauhan et al., 2010; Kanungo et al., 2009; Pokhrel, and Pathak, 2016). It may be noted here that the data regarding the past landslide events in the study area, which occurred in or before 2015, have been used in this work for training the Machine Learning models.

In this study, the Logistic Regression model is implemented with the help of Python programming. Now, Logistic Regression (also called Logit Regression) model, or, to be more specific, Binomial Logistic Regression model, is generally used to estimate the probability that a particular data item belongs to a certain class (Géron, 2017; Laerd Statistics, 2018). If the estimated probability is equal to or greater than 0.5, then the model predicts that the said item belongs to that class (Géron, 2017); otherwise, it predicts that the item does not belong to that class (Géron, 2017). For estimating the probability, the model first computes the sum of the products of the input features and the corresponding weights, and adds the bias term to it (Géron, 2017). Next, it applies logistic function on the result, and the output generated by the logistic function is the estimated probability (Géron, 2017). Equation (1) gives the estimated probability ($\hat{p}$) in mathematical terms:

$$\hat{p} = \frac{1}{1+e^{-(\boldsymbol{\theta}^T \,.\, \boldsymbol{x})}} \tag{1}$$

(Géron, 2017)

where, $\boldsymbol{\theta}$ is the parameter vector of the Logistic Regression model, containing the bias term θ_0, and the weights θ_1 to θ_n, $\boldsymbol{\theta}^T$ is the transpose of $\boldsymbol{\theta}$, $\boldsymbol{x}$ is the input feature vector, containing the features x_0 to x_n, for each data item, with x_0 always equal to 1, and $\boldsymbol{\theta}^T \,.\, \boldsymbol{x} = \sum_{i=0}^{n} \theta_i x_i$ is the dot product of $\boldsymbol{\theta}^T$ and $\boldsymbol{x}$. (Géron, 2017)

(Equation (1) actually represents the logistic function.)

After estimating the probability, the model makes prediction ($\hat{y}$) according to equation (2):

$$\hat{y} = \begin{cases} 0 \;\; if \;\; \hat{p} < 0.5 \\ 1 \;\; if \;\; \hat{p} \geq 0.5 \end{cases} \tag{2}$$

(Géron, 2017)

where, $\hat{y} = 1$ means that the data item, corresponding to the feature vector $\boldsymbol{x}$, belongs to the stipulated (landslide) class, and $\hat{y} = 0$ means that the data item, corresponding to the feature vector $\boldsymbol{x}$, does not belong to the stipulated class. (Géron, 2017)

Before estimating the probability, the optimum values of the weights and the bias term are to be obtained (Géron, 2017). To get these optimum values, this study has minimized the function, depicted in expression (3):

$$\frac{1}{2}\boldsymbol{\theta}^T \,.\, \boldsymbol{\theta} + C \sum_{i=1}^{m} \log\left(1 + e^{-y_i(\boldsymbol{\theta}^T \,.\, \boldsymbol{x}_i)}\right) \tag{3}$$

with respect to $\boldsymbol{\theta}$,

where, m is the number of data items, used for training, y_i is actual output for the ith data item, $\boldsymbol{x}_i$ is the input feature vector for ith data item, and C is a constant. (Fan, Chang, Hsieh, Wang, and Lin, 2008)

It may be stated here that, in expression (3), L2 penalty, in which, the sum of the squares of the parameters, is involved, has been used (Géron, 2017).

The Artificial Neural Network model, implemented in this study with the help of Python programming, is a Multi-Layer Perceptron (MLP) having one input layer with ten neurons (as there are ten input features), excluding the bias neuron, one hidden layer with ten neurons, excluding the bias neuron, and one output layer with one neuron (scikit-learn, 2019; scikit-learn developers, 2019). A bias neuron always yields the value 1 (Géron, 2017; scikit-learn, 2019). For a data item, each input feature, fed to each neuron of the input layer, is multiplied with a suitable weight, and the sum of these products, for all input features, is computed (Géron, 2017; scikit-learn, 2019). This sum is then added to a suitable bias value (multiplied by 1 which is the output of the bias neuron of the input layer), and the resultant sum is fed to a neuron of the hidden layer (Géron, 2017; scikit-learn, 2019). This process is repeated for each neuron of the hidden layer, and the values of the weights and the bias generally differ in each case (Géron, 2017; scikit-learn, 2019). Each neuron of the hidden layer, except the bias neuron, has a Hyperbolic Tangent transfer function, given by equation (4):

$$y = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{4}$$

(Géron, 2017; scikit-learn developers, 2019; Zucker, 1972)

where, x is the input, and y is the output.

The sum of the bias value and the sum of products of the input features and the weights, fed to each neuron, except the bias neuron, of the hidden layer, is actually

provided as input to the transfer function, shown in equation (4), and the output is recorded (Géron, 2017; scikit-learn, 2019). The output of this transfer function, for each neuron of the hidden layer, is multiplied by a suitable weight, and the sum of these products, for all the neurons of the hidden layer, is computed (Géron, 2017; scikit-learn, 2019). This sum is added to a suitable bias value (multiplied by 1 which is the output of the bias neuron of the hidden layer), and the resultant sum is fed to the neuron in the output layer (Géron, 2017; scikit-learn, 2019). This neuron has a Logistic transfer function, given by equation (5):

$$y = \frac{1}{1+e^{-x}} \tag{5}$$

(scikit-learn, 2019)

where, x is the input, and y is the output.

It may be noted here that equations (1) and (5) represent the same function.

The sum of the bias value and the sum of the products of the output values, from the neurons of the hidden layers, and the weights, is provided as an input to the transfer function, shown in equation (5), and the output ($\hat{o}$) is recorded (Géron, 2017; scikit-learn, 2019). In this study, the prediction is made on the basis of this output, as per equation (6):

$$\hat{y} = \begin{cases} 1 & if \ \hat{o} \geq 0.5 \\ 0 & if \ \hat{o} < 0.5 \end{cases} \tag{6}$$

(scikit-learn, 2019)

where, $\hat{y} = 1$ means that the relevant data item belongs to the stipulated (landslide) class, and $\hat{y} = 0$ means that the relevant data item does not belong to the stipulated class. (scikit-learn, 2019)

Before making prediction, optimum values of weights and biases are to be calculated (Géron, 2017). For that, a cost function is required, and the cost function [$f(\boldsymbol{\theta})$], used here, is depicted in equation (7):

$$f(\boldsymbol{\theta}) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^i \log \hat{p}^i + (1-y^i)(1-\log \hat{p}^i)\right] \tag{7}$$

(Géron, 2017; scikit-learn, 2019)

where, m is the number of data items, used for training, y^i is the actual output for the ith data item, $\hat{p}^i$ can be obtained by slightly modifying equation (1), and is given by equation (8):

$$\hat{p}^i = \frac{1}{1+e^{-(\boldsymbol{\theta}^T \cdot x^i)}} \tag{8}$$

where, $\hat{p}^i$ is the estimated probability for the *i*th data item, $\boldsymbol{\theta}$ is the parameter vector, $\boldsymbol{\theta}^T$ is the transpose of $\boldsymbol{\theta}$, and $\boldsymbol{x}^i$ is the input feature vector pertaining to the *i*th data item. (Géron, 2017)

The optimization of the parameter vector, containing the weights and biases, has been done, by employing equation (9):

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \frac{\alpha \widehat{\boldsymbol{m}}_t}{\sqrt{\widehat{\boldsymbol{v}}_t} + \varepsilon} \tag{9}$$

where,

$\boldsymbol{\theta}_t$ is the (more optimized) parameter vector at timestep *t*,

$\boldsymbol{\theta}_{t-1}$ is the (less optimized) parameter vector at timestep *t*-1,

α=0.001 is the stepsize,

$\widehat{\boldsymbol{m}}_t = \frac{\boldsymbol{m}_t}{\sqrt{(1-\beta_1^t)}}$ is the bias-corrected first moment estimate at timestep *t*,

$\widehat{\boldsymbol{v}}_t = \frac{\boldsymbol{v}_t}{\sqrt{(1-\beta_2^t)}}$ is the bias-corrected second raw moment estimate at timestep *t*,

$\boldsymbol{m}_t = \beta_1 \boldsymbol{m}_{t-1} + (1-\beta_1)\boldsymbol{g}_t$ is the biased first moment estimate at timestep *t*,

$\boldsymbol{v}_t = \beta_2 \boldsymbol{v}_{t-1} + (1-\beta_2)\boldsymbol{g}_t^2$ is the biased second raw moment estimate at timestep *t*,

$\boldsymbol{g}_t = \nabla_\theta f_t(\boldsymbol{\theta})$ is the gradient with respect to the cost function [indicated by equation (7)] at time step *t*,

$\boldsymbol{g}_t^2$ is the elementwise square of $\boldsymbol{g}_t$,

β_1=0.9 and β_2=0.999 are the exponential decay rates,

β_1^t and β_2^t denotes β_1 and β_2 to the power of *t* respectively and

ε=10^{-8} is a constant, used to prevent any chance of division by zero.

(Kingma, and Ba, 2015; scikit-learn developers, 2019)

The procedure indicated by equation (9), is repeated until the cost function [equation (7)] is minimized up to a desired level, and an optimum parameter vector is obtained (Kingma, and Ba, 2015).

The Artificial Neural Network model, is also implemented in this study with the help of MATLAB programming. Like the model discussed above, this one is also a Multi-Layer Perceptron (MLP) having one input layer with ten neurons, excluding the bias neuron, one hidden layer with ten neurons, excluding the bias

neuron, and one output layer with one neuron (Beale, Hagan, and Demuth, 2015; Géron, 2017). In this case also, for a data item, the sum of the products of the input features (fed to the neurons of the input layer) and the corresponding appropriate weights, is computed, and a suitable bias value is added to it (Géron, 2017); the resultant sum is fed to a neuron of the hidden layer (Géron, 2017). This process is repeated for each neuron of the hidden layer, except the bias neuron, and the values of the weights and the bias generally differ in each case (Géron, 2017). Here, each neuron of the hidden layer, except the bias neuron, has a Hyperbolic Tangent Sigmoid transfer function, given by equation (10):

$$y = \frac{2}{1+e^{-2x}} - 1 \tag{10}$$

(Beale et al., 2015; MathWorks, 2019)

where, *x* is the input (bias value plus the sum of the products of the input features and the weights), and *y* is the output.

It may be stated here that, equation (10) is equivalent to equation (4).

As done in the case of the previous Artificial Neural Network model, here also, the output of the transfer function [equation (10)], for each neuron of the hidden layer, is multiplied by a suitable weight, and the sum of these products, for all the neurons of the hidden layer, is added to an appropriate bias value (Géron, 2017); the resultant sum is then fed to the neuron in the output layer (Géron, 2017). Now, unlike the previous model, in this case, the neuron in the output layer has a Linear transfer function, given by equation (11):

$$y = x \tag{11}$$

(Beale et al., 2015; MathWorks, 2019)

where, *x* is the input (bias value plus the sum of the products of the output values of the transfer function and the weights), and *y* is the output.

In this study, the output of the transfer function is adjusted so that it lies within 0 and 1 (Géron, 2017). And, the prediction is made on the basis of this adjusted output ($\hat{o}$), as per equation (12):

$$\hat{y} = \begin{cases} 1 \ if \ \hat{o} \geq 0.5 \\ 0 \ if \ \hat{o} < 0.5 \end{cases} \tag{12}$$

(Beale et al., 2015; Géron, 2017)

where, $\hat{y}$ = 1 means that the relevant data item belongs to the stipulated (landslide) class, and $\hat{y}$ = 0 means that the relevant data item does not belong to the stipulated class. (Beale et al., 2015; Géron, 2017)

Like the previous model, here also, optimum values of weights and biases need to be calculated, before making prediction (Géron, 2017); however, the process of finding these optimum values, is different from the previous case. Here, during training, the error in the model's final output, i.e., how the actual output and the desired output differs, is measured (Géron, 2017). Then, the contribution of each neuron in the hidden layer, to this error, is computed (Géron, 2017). Finally, the contribution of each neuron in the input layer, to this error, is calculated (Géron, 2017). The error of each layer is expressed as the MSE or the mean squared error (Beale et al., 2015; MathWorks, 2019). The parameter vector, containing the weights and biases, has been optimized, in this study, by using equation (13):

$$\boldsymbol{\theta}_1 = \boldsymbol{\theta}_0 - (\boldsymbol{J}^T\boldsymbol{J} + \mu\boldsymbol{I})^{-1}\boldsymbol{J}^T\boldsymbol{e} \tag{13}$$

where, $\boldsymbol{\theta}_1$ is the optimized parameter vector, $\boldsymbol{\theta}_0$ is the non-optimized parameter vector, $\boldsymbol{J}$ is the Jacobian matrix, containing the first derivatives of the mean squared errors, with respect to the weights and the biases, $\boldsymbol{J}^T$ is the transpose of $\boldsymbol{J}$, $\boldsymbol{I}$ is the identity matrix of the same dimensions as $\boldsymbol{J}^T\boldsymbol{J}$, $\boldsymbol{e}$ is the vector of mean squared errors, μ is a scalar quantity, used to determine the rate of optimization. (MathWorks, 2019; Sapna, Tamilarasi, and Pravin Kumar, 2012)

The process depicted by equation (13), is repeated until the mean squared errors are minimized up to a desired level, and an optimum parameter vector is obtained (MathWorks, 2019; Sapna et al., 2012).

Next, the output (adjusted output in case of ANN implemented through MATLAB) of LR and ANN models, is translated into suitable Landslide Susceptibility Zonation maps (for map creation, the output/adjusted output of the models is used and not their prediction; prediction is utilized only during testing). The Landslide Susceptibility Zonation maps are expected to help the reader in understanding the zonation information more easily and quickly than the textual descriptions.

A considerable number of studies on Landslide Susceptibility Zonation, have used Artificial Neural Network model; also, quite a number of papers have utilized Logistic Regression model to investigate landslide susceptibility. As for example, (Chauhan et al., 2010) implemented Artificial Neural Network for demarcating the zones of landslide susceptibility in Uttarakhand, and (Pourghasemi et al., 2018) employed Logistic Regression, along with other techniques, for Landslide Susceptibility Zonation in Jumunjin Country, South Korea. The current study has attempted the application of Logistic Regression and Artificial Neural Network (implemented in two ways) models for the Landslide Susceptibility Zonation of a part of the Darjeeling Himalayas. It may be stated here that a number of scientists and researchers, as for example, (Chawla et al., 2018), have already worked on

landslide susceptibility in the Darjeeling Hills, and some of them, like (Mandal, and Mondal, 2019), have utilized Artificial Neural Network and Logistic Regression models. But, work on landslide susceptibility can be done again and again on the same study area, with the hope of getting more accurate results.

2. Materials and Methodology

2.1. Study Area

The bounding box of the study area has the coordinates of 26.974°N latitude and 88.475°E longitude at the north eastern corner, and 26.765°N latitude and 88.234°E longitude at the southwestern corner. The maps showing study area, and its location in the state of West Bengal, are depicted in figures 1a and 1b.

Figure 1a: The study area

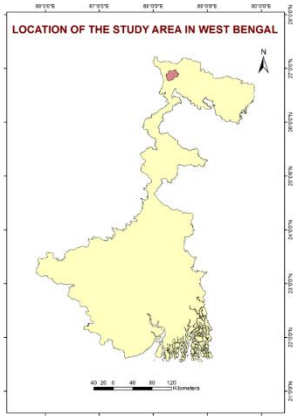

Figure 1b: The study area as a part of West Bengal

2.2. Data

The data are in the form of shapefiles (.shp files), which contain the thematic layers corresponding to land use/land cover, lithostratigraphy, structural features, slope angle, and past landslide events, which occurred in or before 2015. The thematic layers are divided into various classes, whose details are given in table 1.

Table 1: The different classes in the data

Layer	Class
Land use/land cover	**Forest**
	Waterbody
	Built-up

	Agriculture
	Wasteland
Lithostratigraphy	**Daling Group**
	Sikkim Group
	Lingtse Granite Gneiss
	Darjeeling Gneiss
	Siwalik Group
	Gondwana Group
	Quaternary and Recent Sediments
	Damuda Formation (Permian), Chunabhati Formation (Undifferentiated)
	Rangit Pebble Slate, UP Cars – LR Perm, Damuda Formation (Permian) (Undifferentiated)
	Rangit Pebble Slate, UP Cars – LR Perm, Damuda Formation (Permian), Gorubathan Formation (Undifferentiated)
Structural Features	**Within 500m of Lineament**
	Within 500m of Composite Formation
	Within 500m of Fault
	Within 1000m of Thrust
Slope Angle (in Degree)	**< 5**
	5 - 10
	10 - 15
	15 - 20
	20 - 25
	25 - 30
	30 - 35
	> 35
Past Landslide Events	**Nonoccurrence of Landslide**
	Occurrence of Landslide

In table 1, the colours used in the second column, indicate the groupings of different classes as single categories, during processing, considering their contributions to landslide susceptibility. Green indicates low contribution, yellow, medium contribution, deep red, high contribution, and bright red, very high contribution. In case of past landslide events, green indicates a region where no landslide occurred in or before 2015, and deep red indicates a region where landslide/s occurred in or before 2015.

2.3. Method

The outline of the method, employed here, is shown diagrammatically, as a flowchart, in figure 2.

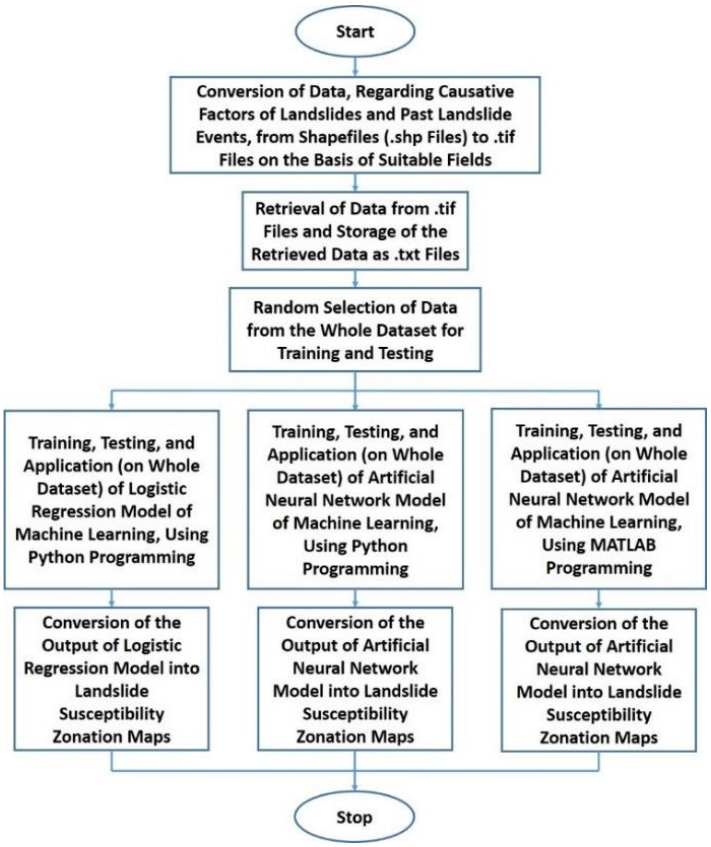

Figure 2: Flowchart

3. Results and Discussions

Before showing the results, the maps depicting the data pertaining to the causative factors of landslides, i.e., the land use/land cover classes, the lithostratigraphy, the structural features and the buffers around them, and the slope angles, and also the data regarding the past landslide events, corresponding to the study area, are shown in figures 3a to 3f.

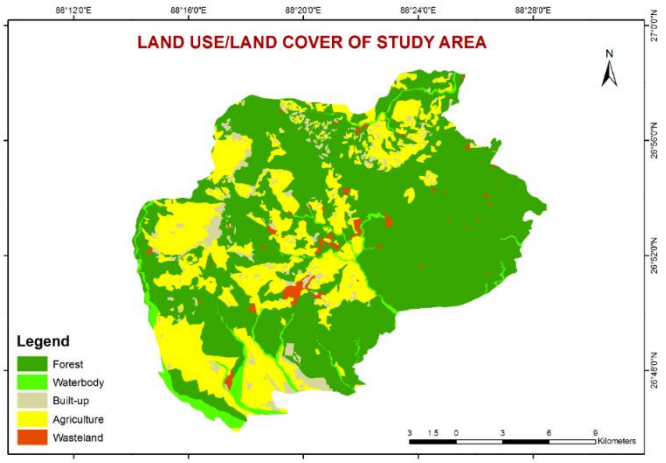

Figure 3a: The land use/land cover of the study area

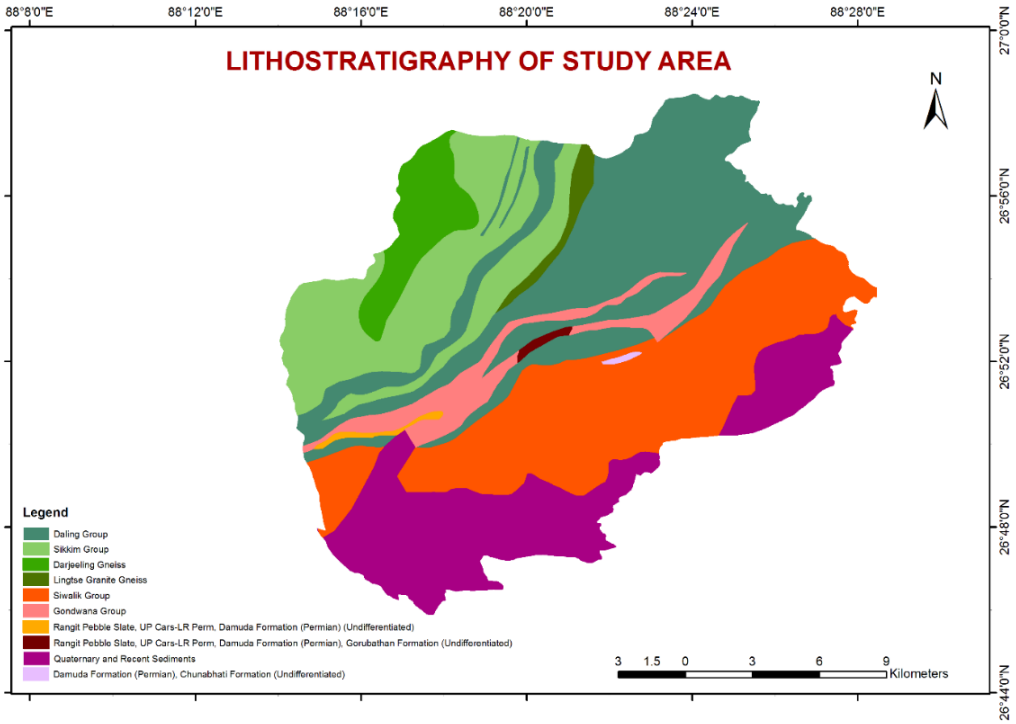

Figure 3b: The lithostratigraphy of the study area (after (Acharya, 1972))

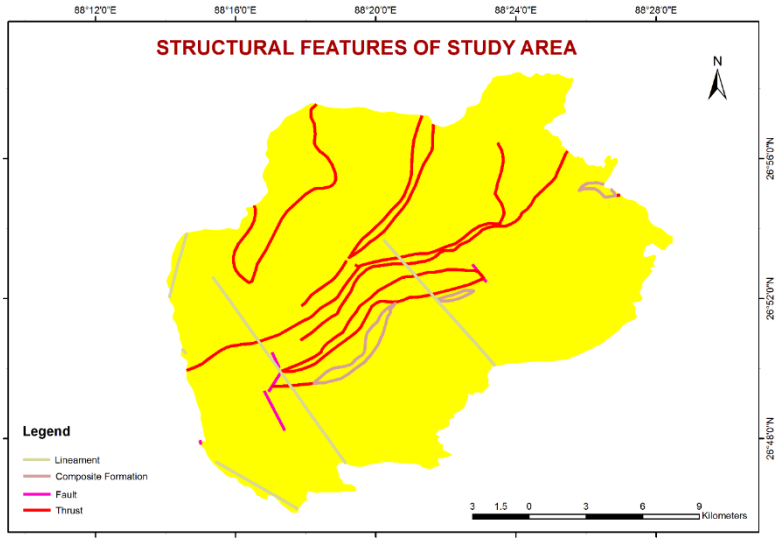

Figure 3c: The structural features of the study area

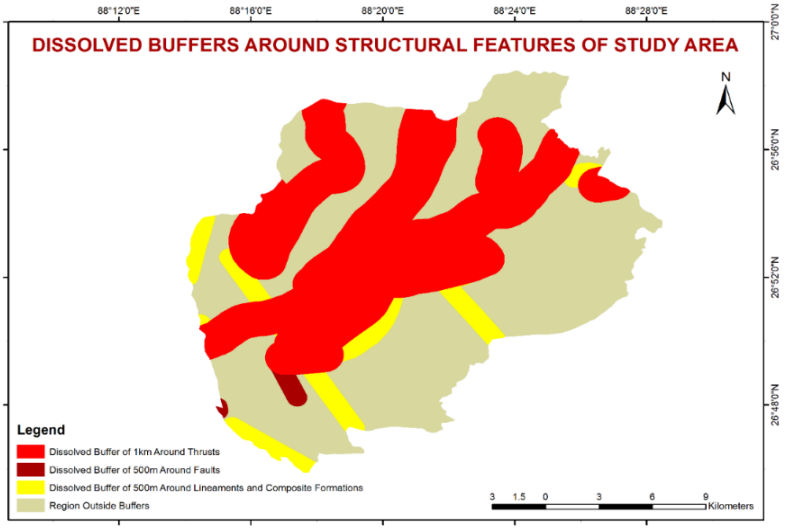

Figure 3d: The dissolved buffers around the structural features of the study area

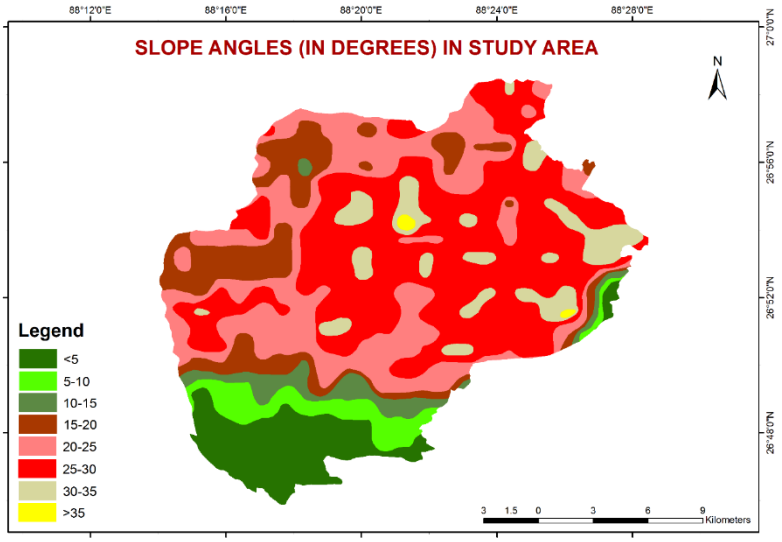

Figure 3e: The slope angles in the study area

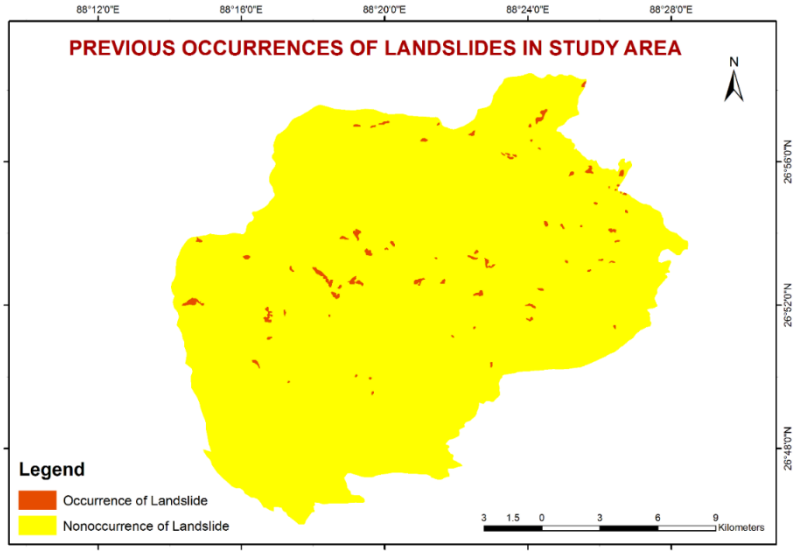

Figure 3f: The past landslide events in the study area

The map of the previous occurrences of landslides, shown in figure 3f, has been prepared through the process of satellite image interpretation, using the DEM (Digital Elevation Model) of the region, for 3D visualization. In this map

(depicted in figure 3f), the landslide occurrence sites are located mostly in the central, northern, and northeastern parts of the study area.

The map, created utilizing Logistic Regression model, which classifies the zones of landslide susceptibility in the study area, using default classes of Reclassify tool of ArcGIS, is depicted in figure 4.

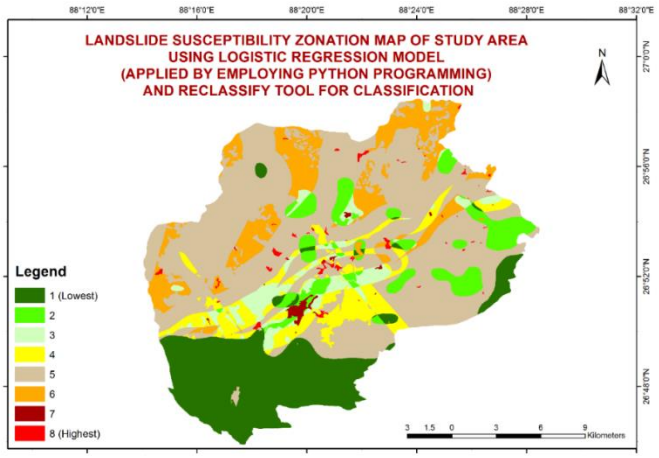

Figure 4: The output from Logistic Regression model, classified using Reclassify tool

In figure 4, the regions with the highest susceptibility, are mostly located in the central, northern, and northeastern parts of the study area.

The map, developed utilizing Artificial Neural Network model (implemented with the help of Python programming), which classifies the zones of landslide susceptibility in the study area, using default classes of Reclassify tool of ArcGIS, is depicted in figure 5.

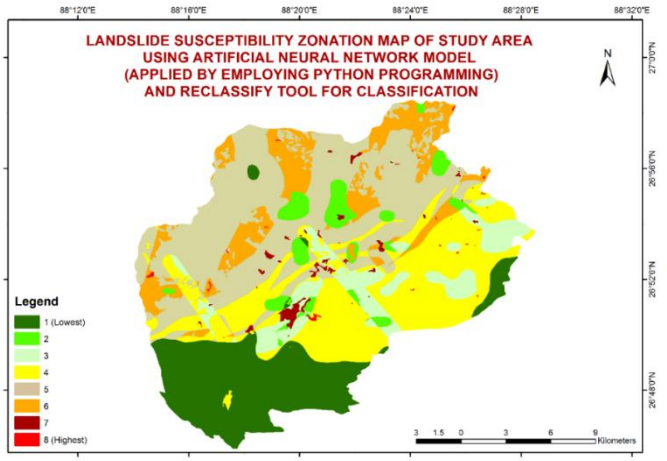

Figure 5: The output from Artificial Neural Network model (implemented by employing Python programming), classified using Reclassify tool

In figure 5, the regions with the highest susceptibility, are mostly located in the central, northern, and northeastern parts of the study area.

The map, developed utilizing Artificial Neural Network model (implemented with the help of MATLAB programming), which classifies the zones of landslide susceptibility in the study area, using default classes of Reclassify tool of ArcGIS, is depicted in figure 6.

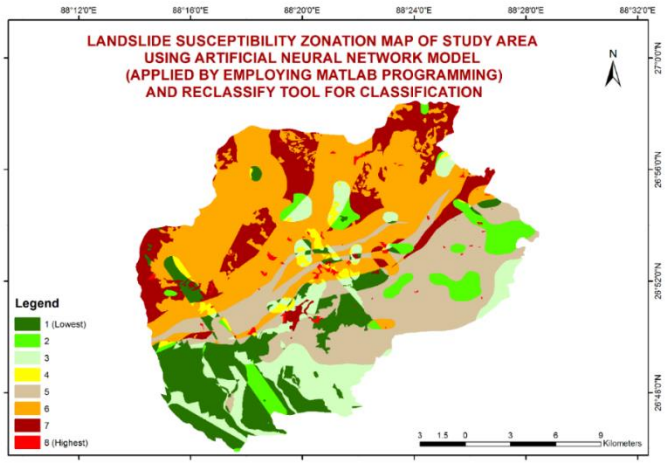

Figure 6: The output from Artificial Neural Network model (implemented by employing MATLAB programming), classified using Reclassify tool

In figure 6, the regions with the highest susceptibility, are mostly located in the central, northern, and northeastern parts of the study area.

Thus, the zones of the highest susceptibility, in the Landslide Susceptibility Zonation maps, generated using Logistic Regression and Artificial Neural Network models, reasonably agree with the sites of the past landslide events, shown in the relevant map. The differences among the three Landslide Susceptibility Zonation maps are probably due to the differences among the models, with regard to the mode of prediction and the technique of optimization.

It has been found that the percentage of error during testing, is 0.7043 for Logistic Regression model, 0.6553 for Artificial Neural Network model (using Python), and 0.6418 for Artificial Neural Network model (using MATLAB). Thus, the models exhibit high levels of accuracy during testing, and that the accuracy of Artificial Neural Network model, implemented using MATLAB programming, is slightly higher than that of either of the other two models (i.e., Logistic Regression model and Artificial Neural Network model, implemented using Python programming).

The novelty of this work is that, with relatively less input (i.e., only four causative factors, and data regarding previous landslide occurrences), it yields Landslide Susceptibility Zonation maps of high accuracy. The technique, used here, may be applicable in those situations where it is hard to get data about many causative factors.

4. Conclusion

Landslide Susceptibility Zonation maps have been created, in this work, for a part of the Kurseong Subdivision of the Darjeeling District of West Bengal, using Logistic Regression and Artificial Neural Network models of Machine Learning; Artificial Neural Network model has been implemented in two ways. For classifying the susceptibility zones, the Reclassify tool of ArcGIS 10.3, with some default classes, has been used. It may be noted that Landslide Susceptibility Zonation of the Darjeeling Hills, has been done before, by some scientists. However, no comparison has been attempted here; only it can be said that, in this study, a high degree of accuracy has been found, while testing, in case of both the models (considering both the implementations of Artificial Neural Network model), and that, the regions of the highest susceptibility, in the Landslide Susceptibility Zonation maps, obtained as the output of this work, reasonably agree with the sites of the past landslide occurrences. Also, it should be mentioned that not much input is demanded by this work.

However, it would have been better, if Logistic Regression and Artificial Neural Network (both the implementations) models were applied in such a way that there is negligible difference among the Landslide Susceptibility Zonation maps, and that these maps agree perfectly with the map of the previous landslide occurrences. If possible, this work can be taken up in future.

The Landslide Susceptibility Zonation maps are expected to help the administrative authority to take preventive measures and/or plan developmental activities in such a way that the loss of human lives, property, infrastructure, and resources, due to landslides, may be avoided, to a great degree.

An advantage of using the Machine Learning techniques is that, the performances of the Machine Learning models, used for processing the data, can be easily tested by crosschecking the output values (obtained during testing) with the target values.

In this study, 20% of the whole dataset has been used for finalizing (i.e., testing and training) the models. And, of this 20% data, 80% has been utilized for training, with the remaining, i.e., 20%, left for testing. The output has been generated by applying each of the optimized Machine Learning models on the whole dataset which includes the data used for training and testing. It may be noted here that, in this work, the sites of past landslide occurrences in the study area are known, and on the basis of these known data, along with the (known) data on the causative factors, the authors want to find the unknown information, i.e., the zones of landslide susceptibility in the same study area. (It has been expected that the zones of highest susceptibility should tally with the sites of past landslide occurrences, and the results reasonably uphold this expectation.) Hence, here, 20% (and not 100%) of the dataset has been utilized for training and testing, and subsequently, the trained and tested models have been applied on the whole dataset. In addition, it may be mentioned here that the data used for training and testing, have been selected randomly. Besides, it should be admitted that there is no particular reason behind selecting the 80-20 scheme; it has been done as per convention only.

In this work, Python programming has been used to implement Logistic Regression model; Artificial Neural Network model has been implemented twice: first time, by employing Python programming, and second time, by utilizing MATLAB programming. Maps have been generated from the output of these models by employing Python programming, and ArcGIS has been used for classification and map composition.

It would have been better, if other models of Machine Learning, e.g., Support Vector Machine (SVM), Naïve Bayes, Convolutional Neural Network (CNN) etc.,

were also used in this study. As for example, Ghosh et al. (2019) used Naïve Bayes model for Landslide Susceptibility Zonation in the hilly parts of Darjeeling and Kalimpong districts. If possible, these (and several other) models can be implemented in future.

Acknowledgment: The authors gratefully acknowledge the kind permission and support received from the Department of Science and Technology and Biotechnology, Government of West Bengal, with regard to this study.

References

[1] Acharya, S. K. (1972). Geology of the Darjeeling Coalfield with Reference to its Intrusives. Records of the Geological Survey of India, 99(2), 23–31.

[2] Beale, M. H., Hagan, M. T., and Demuth, H. B. (2015). MATLAB: Neural Network Toolbox: User's Guide.

[3] Chauhan, S., Sharma, M., Arora, M. K., and Gupta, N. K. (2010). Landslide Susceptibility Zonation Through Ratings Derived from Artificial Neural Network. International Journal of Applied Earth Observation and Geoinformation, 12(5), 340–350.

[4] Chawla, A., Chawla, S., Pasupuleti, S., Rao, A. C. S., Sarkar, K., and Dwivedi, R. (2018). Landslide Susceptibility Mapping in Darjeeling Himalayas, India. Advances in Civil Engineering, Volume 201.

[5] Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). LIBLINEAR: A Library for Large Linear Classification. Journal of Machine Learning Research, 9, 1871–1874.

[6] Géron, A. (2017). Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.

[7] Ghosh, A. R., Parial, K., Mondal, P. P., Kundu, A., Sen, M., and Mitra, D. (2019). Development of a GIS Based Landslide Susceptibility Zonation Model Using Machine Learning Technique in Hilly Parts of Darjeeling & Kalimpong Districts, West Bengal. In Esri India User Conference (UC) 2019. Kolkata.

[8] Kadavi, P. R., Lee, C.-W., and Lee, S. (2018). Application of Ensemble-Based Machine Learning Models to Landslide Susceptibility Mapping. Remote Sensing, 10(8).

[9] Kanungo, D. P., Arora, M. K., Sarkar, S., and Gupta, R. P. (2009). Landslide Susceptibility Zonation (LSZ) Mapping-A Review. Journal of South Asia Disaster Studies, 2(1), 81–105.

[10] Kingma, D. P., and Ba, J. L. (2015). Adam: a Method for Stochastic Optimization. In 3rd International Conference on Learning Representations

(ICLR 2015) Conference Track Proceedings. San Diego. Retrieved from https://dblp.org/db/conf/iclr/iclr2015

[11] Laerd Statistics. (2018). Binomial Logistic Regression Using SPSS Statistics. Retrieved from https://statistics.laerd.com/spss-tutorials/binomial-logistic-regression-using-spss-statistics.php

[12] Mandal, S., and Mondal, S. (2019). Statistical Approaches for Landslide Susceptibility Assessment and Prediction.

[13] Marrapu, B. M., and Jakka, R. S. (2014). Landslide Hazard Zonation Methods: A Critical Review. International Journal of Civil Engineering Research, 5(3 (Special Issue)), 215–220.

[14] MathWorks. (2019). Purelin. Retrieved from https://in.mathworks.com/help/deeplearning/ref/purelin.html

[15] National Oceanic and Atmospheric Administration (NOAA). (2018). What is the Difference Between Land Cover and Land Use. Retrieved from https://oceanservice.noaa.gov/facts/lclu.html

[16] Pokhrel, P., and Pathak, D. (2016). Landslide Susceptibility Mapping of Southern Part of Marsyangdi River Basin, West Nepal Using Logistic Regression Method. International Journal of Geomatics and Geosciences, 7(1), 24–32.

[17] Pourghasemi, H. R., Gayen, A., Park, S., Lee, C.-W., and Lee, S. (2018). Assessment of Landslide-Prone Areas and Their Zonation Using Logistic Regression, LogitBoost, and NaïveBayes Machine-Learning Algorithms. Sustainability, 10(10).

[18] Sapna, S., Tamilarasi, A., and Pravin Kumar, M. (2012). Backpropagation Learning Algorithm Based on Levenberg Marquardt Algorithm. Computer Science & Information Technology, 2(4), 393–398.

[19] scikit-learn. (2019). 1.17. Neural Network Models (Supervised). Retrieved from https://scikit-learn.org/stable/modules/neural_networks_supervised.html

[20] scikit-learn developers. (2019). scikit-learn: scikit-learn User Guide (Release 0.21.3).

[21] Zhou, C., Yin, K., Cao, Y., Ahmed, B., Li, Y., Catani, F., and Pourghasemi, H. R. (2018). Landslide Susceptibility Modeling Applying Machine Learning Methods: A Case Study from Longju in the Three Gorges Reservoir Area, China. Computers and Geosciences, 112, 23–37.

[22] Zucker, R. (1972). Elementary Transcendental Functions: Logarithmic, Exponential, Circular and Hyperbolic Functions. In M. Abramowitz & I. A. Stegun (Eds.), Handbook of Mathematical Functions With Formulas, Graphs, and Mathematical Tables (p. 83).