Predicting Malarial Outbreak using Machine Learning and Deep Learning Approach: A Review and Analysis

Conference Paper · December 2018

DOI: 10.1109/ICIT.2018.00019

3 authors, including:

Godson Koffi Kalipe
KIIT University
2 PUBLICATIONS **47** CITATIONS
SEE PROFILE

Rajat Kumar Behera
KIIT, Deemed to be University
28 PUBLICATIONS **219** CITATIONS
SEE PROFILE

Predicting malarial outbreak using Machine Learning and Deep Learning approach: A review and analysis

Godson Kalipe
School of Computer Engineering
KIIT Deemed to be University

Vikas Gautham
School of Computer Engineering
KIIT Deemed to be University
Bhubaneswar, India
vgautham99@gmail.com

Rajat Kumar Behera
School of Computer Engineering
KIIT Deemed to be University
Bhubaneswar, India
rajat_behera@yahoo.com

***Abstract*—In the present era of information, data has revealed itself to be more valuable to organizations than ever before. By applying machine learning and deep learning approaches to historical or transactional data, we are now able to derive new ground breaking insights helping us to make better informed decisions and adopt the best strategies in order to face the events that are likely to happen in the future. In this paper, we have not only sought to establish a relationship between climatic factors and a possible malarial outbreak but we also tried to find out which algorithm is best suited for modeling the discovered relationship. For that purpose, historical meteorological data and records of malarial cases, collected over six years, have been combined and aggregated in order to be analyzed with various classification techniques such as KNN, Naive Bayes, and Extreme Gradient Boost among others. We were able to find out few algorithms which perform best in this particular use case after evaluating for each case, the accuracy, the recall score, the precision score, the Matthews correlation coefficient and the error rate. The results clearly implied that weather forecasts could be legitimately leveraged in the future to predict malarial outbreaks and possibly take the necessary preventing measures to avoid the loss of lives due to malaria.**

Keywords— Classification algorithms, Machine Learning, Deep Learning, Malaria prediction

I. Introduction

Malaria is mosquito-borne disease which has been for several years one of the main causes of death around the world. In the year 2016 only, 216 million cases were identified leading to 445 000 deaths [1]. India is the third most affected country worldwide with 6% of all the cases identified within the country [1]. It is well established that climate plays a major role in the proliferation of malaria since wet and warm environments are more suitable for the breeding of the mosquitoes responsible for its transmission [2]. This work tries to determine how helpful various climatic indicators can be in predicting whether or not a malarial outbreak would occur. Several classification algorithms are tested for that purpose and we try to find out which one is most suitable for the prediction in this particular use case.

Deep learning is a subset of machine learning techniques which themselves belong to the bigger framework of Artificial Intelligence. These techniques provide ways for computers, given data, to be able to learn and get better at how to perform a particular task without explicitly being programmed to.

Here, we first present the data we used and we describe the methodology that was used to extract valuable information from it. Then, we briefly introduce each of the algorithms that we have used. We continue by defining the different performance indicators used. Finally, we offer the major results of our work followed by their possible implications before finishing with the conclusion.

II. Data Sources

We have used over 6 years of data collected from various health centers in the district of Visakhapatnam and aggregated per mandal between 2005 and 2011. That data includes information about the two most popular types of malaria: the ones caused by the plasmodium falciparum (pf) and the ones caused by the plasmodium vivax (pv). The total population per mandal is also recorded in the same dataset. This first dataset was produced by the National Vector Borne Disease Control Program. We combined that data with climate data observed over the same period of time with respect to each mandal provided by Indian meteorological Centre and Cyclone Warning Center. Both datasets were retrieved from the supporting information section of [3].

III. Methodology

We start by defining the problem which is to be able to classify the various mandals in the district of Andhra Pradesh in two classes with respect to whether or not a malarial outbreak is likely to happen there, based on the number of cases of malaria detected and some other atmospheric factors.

A. Data Cleaning and transformation

Once the data has been collected in the format of csv files, we have extracted from it the information related to the year 2005. We have divided the number of cases of malaria (due to both plasmodium falciparum and vivax) by the total amount of the population in a given mandal to obtain the ratios. The ratios of cases caused by the two different kinds of plasmodium were summed to generate a total ratio value for each mandal per month. The range of values assumed by the total ratio features has then divided into three equal intervals. They were obtained using the equation (1).

$$\text{diff} = \frac{\max\_ratio - \min\_ratio}{3} \ldots \qquad (1)$$

We then defined 3 levels of risk:

Low, for mandals having a total ratio value in the interval min_ratio and min_ratio + diff, for a given month

Medium, for mandals having a total ratio value in the interval min_ratio + diff and min_ratio + 2 x diff, for a given month

High, for mandals having a total ratio value in the interval min_ratio + 2 x diff and max_ratio for a given month

Finally we added to the dataset the feature "Outbreak" as dependent variable. Its values can be "Yes" if risk is "Medium" or "High" or "No" if risk has the value "Low".

B. Feature Selection

Before moving to the application of our various classification algorithms, we selected the following features as our variables :

Outbreak, the dependent variable, to be predicted using the values of the other independent variables. Our problem is thus a two-class classification problem.

MinTemp, the independent variable which represents the minimum temperature noted in a particular mandal in a particular month

MaxTemp, the independent variable which represents the maximum temperature noted in a particular mandal in a particular month

Humidity, representing the measured humidity level

Total_pop_ratio, the independent variable representing the ratio of the number of malaria cases by the population of a given mandal.

C. Execution

The dataset was divided in two parts: A sample containing 80% of the data for the training and 20% for testing purpose. Then, the models were trained on the training sample using 7 main classification algorithms implemented in Python: KNN, Random Forest, Support Vector Machine, Extreme Gradient Boosting, Logistic regression, Neural network and Naive Bayes. The resulting models were tested on the 20% remaining data and the results were compared with the initial values of the Outbreak feature in the original dataset.

IV. ALGORITHMS USED

We have used and compared the performance of some of the most popular classification algorithms both classic and recent.

A. Linear Regression

Regression is a technique used to predict the value of a dependent variable based on the values of independent numeric variables. There exist multiple forms of regression. Linear regression tries to find a line that best represents the statistical relation between the predictors and target features. The line's equation is described in equation (2):

$$Target = a * predictor + b \ldots \quad (2)$$

a represents the slope of the line and *b*, the intercept of the same line [13].

B. Logistic Regression

This algorithm is used when the dependent feature is binary in nature. In the case of logistic regression, there is no need to establish a linear relation between the independent variables and the dependent one.

The same formula is used for logistic regression but the sigmoid function is applied to the result in order to keep the final result between 0 and 1. The formula is described in equation (3) where Y is expressed as the target of the linear regression's function [13].

$$\text{diff} = \frac{1}{1 + e \exp - Y} \ldots \quad (3)$$

C. Naive Bayes

It is a probability based classifier that makes the assumption that all the features used for prediction are independent from each other. Based on the training sample data it computes probability of each of the values of the dependent variable corresponding to each combination of the values (or ranges of value) possibly assumable by the independent variables [14]. The probability of a value Y of the target variable, given an array F of values of dependent values is obtained using the following formula in equation (4) :

$$\mathrm{P}\left(\frac{\mathrm{Y}}{\mathrm{F}}\right) = \frac{\mathrm{P}\left(\frac{F}{Y}\right) * \mathrm{P(Y)}}{\mathrm{P(F)}} \ldots (4)$$

The other important formula this algorithm uses is:

$$P(Y \,/\, F1,F2,F3 \ldots . Fn) = \mathrm{P}\left(\frac{\mathrm{F1}}{\mathrm{Y}}\right) * \mathrm{P}\left(\frac{\mathrm{F2}}{\mathrm{Y}}\right) * \ldots . * \mathrm{P(Fn \,/\, Y)} * \mathrm{P(Y)} \ldots \quad (5)$$

In the above equation, F1…FN represents the n features used in the problem. For example, in our case, considering the feature humidity, it can assume values between 50 and 80. The algorithm could first calculate for instance the probabilities that the humidity is in the following respective intervals between 50 and 60, between 60 and 70 or between 70 and 80 when an outbreak happens. Then it will calculate the same probabilities if an outbreak doesn't happen. Finally, it will compute the overall probabilities that an outbreak happens or not independently from any factor. A similar process happens for every feature.

The resulting probabilities are combined using the Naive Bayes formula:

$$P\,(target = y1/F1 = val1, F2 = val2 \ldots . Fn = valn) = P\,(F1 = val1 \,/\, target = y1) * P(\,F2 = val2 \,/\, target = y1) * \ldots * P(Fn = valn \,/\, target = y1) * P(target = y1) \ldots (6)$$

The above formula is used to predict the probability of a value of a target variable given an array of values of features determining it.

D. Support Vector Machine (SVM)

This algorithm also known as Support Vector Classifier (SVC) computes the equation of a hyperplane to separate groups of points in two distinct classes [15]. In two dimensions (a two-feature classification problem), the hyperplane consists of a simple line. The line can be distorted to achieve the best accuracy (or to separate non linearly separable points) by using kernel transformations. Sometimes, that transformation happens at the expense of the training time. There exist two parameters known as "gamma" and "regularization parameter" that are used to tune the algorithm and obtain the best compromise between execution time and accuracy of a non-linear classification. The first one determines the size of the margin of the separating hyperplane and the second one determines the distance limit of the points considered in the calculation of the hyperplane.

E. K Nearest Neighbors (KNN)

It is a famous classification technique known to provide easy to interpret output, reasonable calculation time and strong predictive power. It is a similarity based algorithm that associates to each new element the most common class among the elements closest to this one (the k nearest element) [6]. The trickiest part remains the choice of the "k" argument in order to minimize both the training's and validation step's error rates.

F. eXtreme Gradient Boosting (XGBoost)

It is focused on model performance and computational speed (without sacrificing one over the other). It is particularly popular among the online competition data science community because of its rapid and accurate results that have helped many win competitions in the past years [7]. It is based on Gradient Boosting machine algorithms which purpose is to combine various weak machine learning models to achieve stronger ones. Other examples include the famous AdaBoost. In the approach adopted by XGBoost, new models are created to predict the errors or residuals of previous models.

It is able to solve ranking, regression and classification problems.

G. Random Forest

A random forest is a collection of many decision trees built by choosing a random attribute for splitting at each node [8]. The class that is most frequently predicted across all trees in the forest is chosen as the predicted class by the random forest. It helps to reduce over-fitting by allowing the detection of attributes which are contributing less or nothing to the prediction of the outcome and by building several trees. (It is well-known that a huge number of features in a decision tree contribute to over-fitting). We can tune the algorithm to have more trees which increase its performance but also its execution time. We can also tune the maximum number of features each individual tree can use. They can handle binary, categorical and numerical data types.

H. Neural networks

To build our model, we have used the open source python library keras. It is probably the most popular high-level deep learning library for Python. It wraps two other low-level popular libraries Theano and TensorFlow and can use any of them as backend. It also greatly simplifies the neural network building process [12].

V. RESULTS AND IMPLICATIONS

A. Algorithms comparison factors

We have used several comparison metrics to compare between the machine learning algorithms used in order to determine not only which one performs best in this particular use case but also which one is likely to perform equally good provided a similar dataset of bigger size. In this section we introduce and define the meaning of those metrics and explain why we used them [10, 11].

1) Accuracy

It is the most straightforward and most common way of measuring the performance of an algorithm.

$$\text{Accuracy} = \frac{Number\ of\ correct\ predictions\ made}{\text{Total number of predictions made}} \ldots \quad (7)$$

There are some types of problems where accuracy has proven to be insufficient and misleading in helping assess the performance of an algorithm. It is the case for imbalanced problems where one target variable value has way more occurrences than the other. When the cost of classification rises, we cannot afford to rely only on accuracy. Other metrics that can solve that limitation are introduced in the next subsection [10].

Before we define the rest of the metrics used, we need to define the following terms that will be used in the formula given in the next section :

True Positives (TP): It is the number of positive target variable instances that have been correctly predicted as positive

False Positives (FP): It is the number of negative target variable instances that have been wrongly predicted as positive

True Negatives (TN): It is the number of negative target variable instances that have been correctly predicted as negative

False Negatives (FN): It is the number of positive target variable instances that have been wrongly predicted as negative.

2) Recall

Also called sensitivity, it evaluates how many positive targets are lost in the predication.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \ldots \quad (8)$$

3) Precision

Also called positive predictive value, it helps evaluate how many positives are superfluous in the result and thus cannot be trusted.

$$\text{Precision} = \frac{TP}{\text{TP} + \text{FP}} \ldots \quad (9)$$

4) Error Rate

The error rate (ER) is the metric opposite to the accuracy. It is obtained by the following formula :

$$\text{ER} = \frac{FN + FP}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \ldots \quad (10)$$

It evaluates in percentage the number of instances that are being wrongly classified by the model.

5) Matthews Correlation Coefficient

The matthews correlation coefficient (MCC) is obtained using the following formula:

$$MCC = \frac{\text{TP}*\text{TN} - \text{FP}*\text{FN}}{\sqrt{(TP+FP)*(FP+TN)*(FP+TN)*(TP+FN)}} \ldots \quad (11)$$

It can take values in the interval between -1 and +1 where -1 indicates a classifier that predict opposite classes, 0 a classifier that predicts randomly and +1 a classifier that predicts perfectly.

[11] describes how well this metric can be leveraged to detect inefficient classifiers even when the accuracy and recall values might be good.

B. Implementation specifications

The code was written in Python using the sklearn library which constitutes an almost complete toolbox for machine learning in Python.

The data was initially read from the csv file into a dataframe. The outbreak feature was then encoded in numeric format. After the independent variables matrix was separated from the dependent variable vector, two third of the dataset was extracted to train the models and one third was kept to test their performance.

Finally, the data in both the training and testing set were scaled to values in the interval between 0 and 1in order to improve the efficiency of the training process and minimize the impact of high numerical differences between the ranges of the different independent variables.

Our neural network was made of a sequential combination of two layers of neurons having respectively a hyperbolic tangent and sigmoid function. Outputs greater than 0.5 were considered true and outputs below that threshold were considered false.

C. Results of the implementation

The overall results of all the algorithms have been recorded, summarized and organized in table 1.

TABLE I. TABLE SUMMARIZING THE PERFORMANCES OF THE VARIOUS ALGORITHMS

Algorithms	Accuracy (%)	Recall (%)	Precision (%)	MCC (%)	Error Rate (%)	Specificity (%)	FPR (%)
KNN	86.21	50.48	92.9	61.92	13.79	98.66	1.34
Random Forest	93.94	92.28	85.42	84.7	6.06	94.51	5.49
SVM	92.69	79.42	91.14	80.42	7.31	97.31	2.69
XGboost	96.26	93.89	91.82	90.33	3.74	97.09	2.91
Logistic Regression	92.44	79.1	90.44	79.74	7.56	97.09	2.91
ANN	25.83	100.0	25.83	0.0	74.17	0.0	100.0
Naïve Bayes	91.69	78.14	88.36	77.73	8.31	96.42	3.58

XGBoost, ANN, Random Forest, SVM perform best achieving respectively the highest number of correctly predicted outbreak values. Fig. 2 represents the percentages of correct predictions each of the algorithms achieved in blue compared to the percentages of wrong predictions in orange.

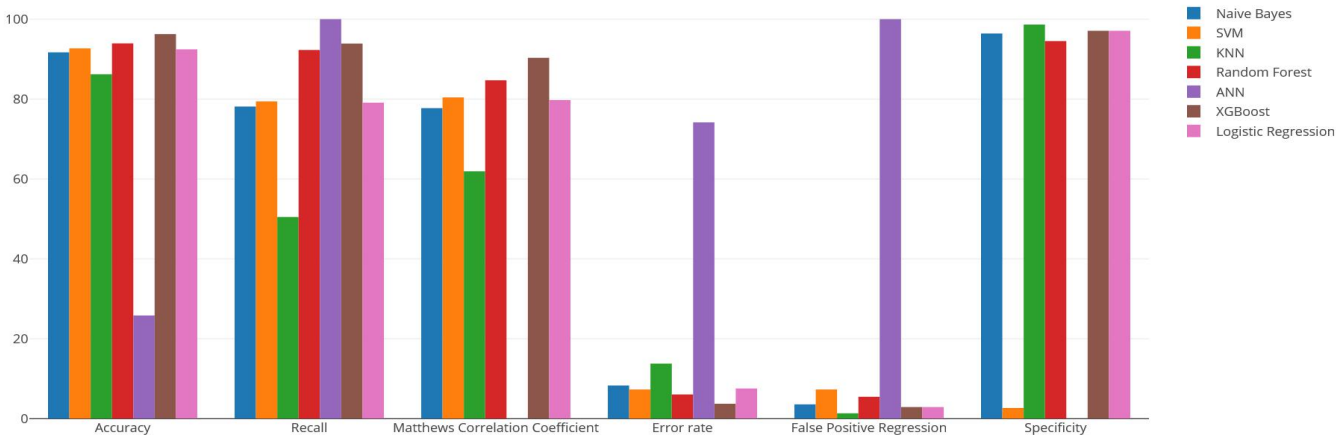

Fig. 1. Comparison of all the performance metrics for the different algorithms

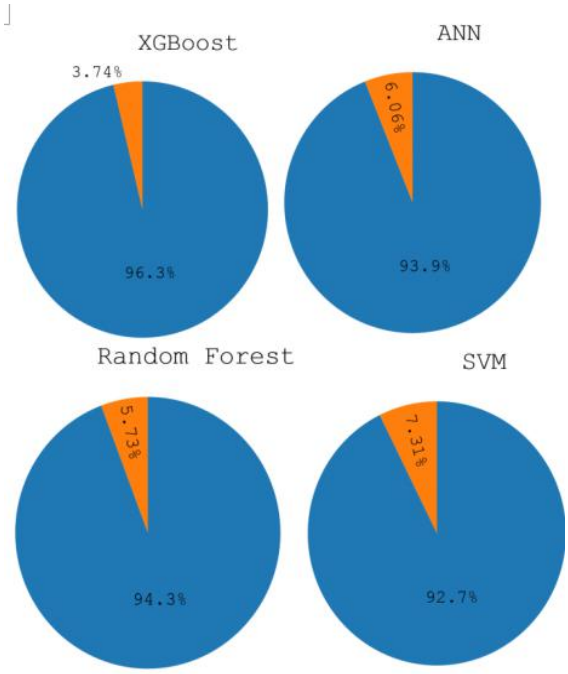

Fig. 2. Percentages of correct and wrong predictions for the four best performing algorithms

A comparison of the different performances was plotted on a bar chart available in Fig. 1. One can observe that XGBoost clearly outperforms all of its other competitors not only in terms of accuracy but also in terms of precision and recall for this particular use case. It achieves an accuracy of 96.26% of correctly predicted values backed up by a solid recall of 93.89%. The precision is the lowest with 91.82% achieved. It means XGBoost is more likely to predict false positive outbreak values. Thus, the precision can be improved in order to avoid unnecessary emergency alerts due to the wrong predictions of the model.

VI. CONCLUSION

The results obtained by this work show that the atmospheric factors and recorded number of cases of malaria disease can be effectively used to predict whether or not a malarial outbreak would occur. These results also imply that the XGBoost algorithm is particularly efficient for this particular use case. Nevertheless the weak precision value ought to be improved as the problem that is dealt with here is very sensitive and involves many lives at stake on a big scale. In the future, this work can be extended by combining several of these algorithms to further improve the reliability of the model built. It is also possible to add other atmospherical factors to the independent variable matrix in order to refine the predicting capability of the model.

REFERENCES

[1] 'World Malaria Report 2017,' WHO, Geneva, Greece, Rep. ISBN 978-92-4-156552-3, 2017.

[2] P. Rekacewicz, 'Climate Change and Vector-Borne Disease', NCAR UCAR Science Education. 2011 [Online]. Available : https://scied.ucar.edu/longcontent/climate-change-and-vector-borne-disease . [Accessed: 20 - sep - 2018]

[3] R. C. P. K. Srimath-Tirumula-Peddinti, N. R. Neelapu, and N. Sidagam, 'Association of Climatic Variability, Vector Population and Malarial Disease in District of Visakhapatnam, India: A Modeling and Prediction Analysis,' PLoS One, vol. 10, no. 6, Jun. 2015. [Online]. Available: https://dx.doi.org/10.1371%2Fjournal.pone.0128377

[4] B. H. Malini, B. V. Reddy , M. Gangaraju and K. N. Rao, 'Malaria risk mapping : a study of Visakhapatnam district,' Current Science, vol. 112, no. 3, pp. 463-465, Feb. 2017. [Online]. Available : https://www.researchgate.net/publication/313528185_Malaria_risk_mapping_a_case_of_Visakhapatnam_district

[5] 'Deep learning,' Wikipedia. 2018 [Online]. Available: https://en.wikipedia.org/wiki/Deep_learning . [Accessed: 20 - sep - 2018]

[6] T. Srivastava, 'Introduction to k-Nearest Neighbors: Simplified (with implementation in Python)', Analytics Vidhya. 2018 [Online]. Available: https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/ . [Accessed: 19 - sep - 2018]

[7] J. Brownlee, 'A Gentle Introduction to XGBoost for Applied Machine Learning', Machine Learning Mastery. 2016 [Online]. Available : https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/ . [Accessed: 19 - sep - 2018]

[8] N. Donges, 'The Random Forest Algorithm', Towards Data Science. 2018 [Online]. Available : https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd . [Accessed: 19 - sep - 2018]

[9] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[10] W. Koehrsen, 'Beyond Accuracy : Precision and Recall', Towards Data Science. 2018 [Online]. Available : https://towardsdatascience.com/beyond-accuracy-precision-and-recall-3da06bea9f6c . [Accessed: 19 - sep - 2018]

[11] D. Lettier, 'You need to know about the Matthews Correlation Coefficient', Content by David Lettier. [Online]. Available :

https://lettier.github.io/posts/2016-08-05-matthews-correlation-coefficient.html . [Accessed: 22 - sep - 2018]

[12] 'Why use Keras', Keras Documentation. 2018 [Online]. Available : https://keras.io/why-use-keras/ . [Accessed: 22 - sep - 2018]

[13] 'Difference Between Linear and Logistic Regression', TechDiffrences , [Online], Available : https://techdifferences.com/difference-between-linear-and-logistic-regression.html . [Accessed : 28 - sep - 2018]

[14] S. Patel, 'Supervised Learning and Naive Bayes Classification', Machine Learning 101, [Online], Available : https://medium.com/machine-learning-101/chapter-1-supervised-learning-and-naive-bayes-classification-part-1-theory-8b9e361897d5 . [Accessed : 28 - sep - 2018]

[15] S. Patel, 'Chapter 2 : SVM (Support Vector Machine) - Theory', Machine Learning 101, [Online], Available : https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72 . [Accessed : 28 - sep - 2018]