Forest Fire Prediction using Machine Learning Techniques

1st Therese Taylor
ITRI 616
North-West University
Potchefstroom, South Africa
25830473@mynwu.ac.za

2nd Gian van Zijl
ITRI 616
North-West University
Potchefstroom, South Africa
38098806@mynwu.ac.za

***Abstract*—Forest fires occur frequently in South Africa, causing substantial damage and threatening lives, property, and the environment. Early prediction of these fires is therefore crucial to minimise the damage they cause. Few studies use meteorological data to predict the likelihood of fire occurrence, despite meteorological and climatic conditions being the primary factors in the ignition and spread of fires. Most existing research focusses on detection after the fire has already started, when predicting fire likelihood in advance could help to proactively mitigate the threat. This study looks at whether machine learning (ML) techniques can predict forest fires before they occur using meteorological data. This will be done by comparing the results of three ML techniques: Random Forest (RF), Support Vector Machine (SVM), and K-Nearest Neighbours (KNN).**

***Index Terms*—forest fires, meteorological data, machine learning**

I. Introduction

Forest fires occur frequently in South Africa, causing substantial damage and threatening lives, property, and the environment [1]. Over 40,000 fires are reported yearly, resulting in nearly R4 billion in financial losses. The most devastating wildfires to date occurred in June 2017, affecting the Garden Route in the Southern Cape. These fires resulted in seven deaths, the destruction of more than 1,000 structures, and damage to 500 houses. Thousands were evacuated, with 1,533 families and 134 businesses directly impacted. Essential infrastructure, including power lines, was also damaged or destroyed. The primary causes of recurring forest fires include dry seasonal climates, natural vegetation that provides fuel, and fires ignited by humans. Hot, dry, and windy conditions, along with accumulated fuel, contribute significantly to fire damage. Current development trends also increase fire risks. With the rapid growth of the population and high urbanisation rates, more people, assets and infrastructure are located at the boundary between developed areas and fire-prone vegetation, increasing their exposure to these forest fires. Early prediction of these fires is therefore crucial to minimise the damage they cause. Several methods and technologies have been proposed for this purpose, but recently the trend is toward the utilisation of machine learning techniques, as they have shown promise in predicting the occurrence and spread of forest fires.

A survey on the use of machine learning algorithms for forest fire detection and prediction [2] revealed that most of the relevant literature has concentrated on vision-based fire detection using images or videos, predicting human-caused fire occurrences, and using WSN technology for real-time detection and AI integration. Some studies have also looked at predicting burnt areas using meteorological data and fire indices. However, there are few studies that use meteorological data to predict the likelihood of fire occurrence, despite the survey indicating that meteorological and climatic conditions are primary factors in the ignition and spread of fires. Most existing research focusses on detection after the fire has already started, when predicting fire likelihood in advance could help to proactively mitigate the threat.

Therefore, this study looks at whether machine learning (ML) techniques can predict forest fires before they occur using meteorological data. This will be done by comparing the results of three ML techniques: Random Forest (RF), Support Vector Machine (SVM), and K-Nearest Neighbours (KNN). Section II provides

a discussion on the selection of techniques used to address this problem.

II. Choice of Techniques

The task of predicting forest fires can be framed as a classification problem, where the goal is to determine whether a fire is likely to occur based on observed data [3]. These problems can be defined where $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ is a set of observations having q-dimensional patterns. The data $X = \{x\}_{i=1}^{N} \subset \mathbb{R}^q$ have a corresponding set of labels $Y = \{y_i\}_{i=1}^{N} \subset \mathbb{R}^d$. Classification models aim to predict the class label y' for an unknown pattern x' using a functional model f. We thoroughly evaluated several widely recognised algorithms commonly used in classification tasks. After careful consideration, we chose three specific ML algorithms for analysis: RF, SVM, and KNN. Each of these algorithms offers distinct advantages and characteristics that make them suitable for different types of classification problems, providing a robust foundation for our analytical approach. Section III includes a discussion on the dataset and the data preparation process.

III. Dataset description and preparation

There were no data sets specific to South Africa available, so we obtained a data set from the UC Irvine Machine Learning Repository that contains data from two regions of Algeria [4]. The data is collected during the summer of 2012, as this was a period with high fire occurrences. It comprises 244 entries, with 138 instances labelled ”fire” and 106 instances labelled as ”not fire.” Originally, the data set included 14 attributes, such as date attributes, temperature, relative humidity, wind speed, rain, components of the Canadian Fire Weather Index, and the class label. During the data preparation phase, we excluded the date attributes since the data collection period was too short to have a significant impact. The components of the Canadian Fire Weather Index were also removed as they were used solely to generate class labels during data set creation. This left four crucial weather elements: temperature (in Celsius), relative humidity (as a percentage), wind speed (in km / h) and rain (mm/m^2). In addition to this, the class labels were changed to numerical values (fire = 1 and not fire = 0) and the data set was cleaned to address issues such as null values, missing spaces, and empty spaces. Section IV contains a detailed discussion on the modeling, simulation experiments, and program design.

IV. Modelling, Simulation Experiment and Program Design

In this section, we critically evaluate the models, discussing their specific data splits and hyperparameter settings. Additionally, we present the results for each model, along with a brief discussion, including insights from k-fold cross-validation. Subsections A, B, and C will respectively explore the RF, SVM, and KNN models.

A. Random Forest

Random forest is a decision tree ensemble; an ensemble technique combines predictions from various different models to produce one final result. This means that a random forest consists of multiple individual decision trees. Each of these trees consists of a randomly chosen subset of data points and features within the training data set, reducing the correlation among the trees. The prediction made by each tree is aggregated to reach the final prediction result [5]. This process is demonstrated in Figure 1. The model was built using the RandomForestClassifier algorithm from the scikit-learn in Python [6]. To determine the best evaluation method, we implemented an 80/20 split, with 80% for training and 20% for testing, as well as a 5-fold cross-validation setup (with a holdout set of 20%). The training and test accuracy of these methods is compared in Table 1.

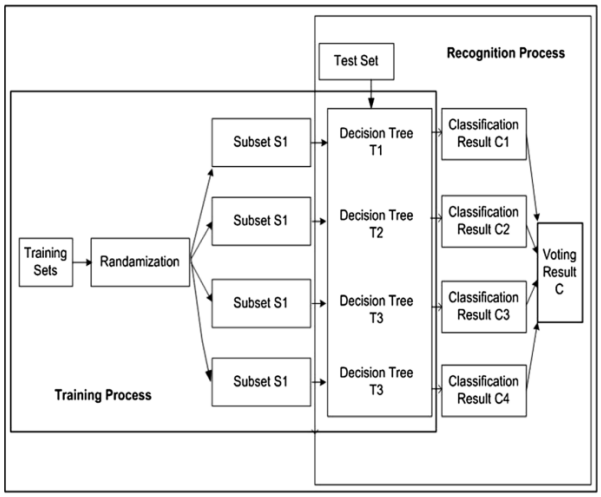

Fig. 1. Structure of a random forest classifier [5].

TABLE I
COMPARISON OF EVALUATION METHODS

Method	Training accuracy	Testing accuracy
80/20 split	89%	86%
Cross-validation	86%	85%

The 80/20 split method performs better for both training and testing, which is expected, as cross-validation is generally seen as more reliable and uses a smaller part of the already small data set in this case. However, since the 80/20 method performed better, it will be used for the rest of the study. To improve the performance of the model after the initial assessment, we used the feature importance attribute. It indicated that rain and temperature are the most significant features. Consequently, we excluded wind speed and relative humidity, resulting in an improvement in the algorithm's accuracy. We also fine-tuned the hyperparameters to further improve the model's performance, with a final hyperparameter setting as follows: n estimators=50, max depth=5, min samples split=12, min samples leaf=1. The model's classification report can be seen in Table 2.

TABLE II
CLASSIFICATION REPORT

	precision	recall	f1-score
not fire	95%	77%	85%
fire	79%	96%	86%

This classification report shows that the model performs reasonably well, with high precision, recall, and F1 scores for both classes. However, in general, the model is better at correctly identifying instances of fire (Class 1) compared to instances of not fire (Class 0). This suggests a bias towards false positives, which could lead to a higher risk of false alerts in practical applications. This can be further supported by the confusion matrix (Figure 2).

B. SVM

SVM is a kernel-based ML model for performing regression and classification tasks [7]. This model has demonstrated its efficacy in tackling practical binary classification tasks. Training these models involves utilising a dataset comprising n examples, each composed of an input vector x_i and its corresponding label y_i. The dataset X can be represented as $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, where $X = \{x_i, y_i\}_{i=1}^{n}$, $x_i \in \mathbb{R}^d$, and $y_i \in \{-1, 1\}$. Numerous hyperplanes can adeptly segregate linearly separable data shown in Figure 3. The positioning and orientation of the optimal separation hyperplane profoundly impact the model's ability to generalise. This hyperplane, maximising the margin, is pivotal in delineating the decision boundary within the input space, defined by the equation $w^T x_i + b = 0$. In the case of linearly separable features, the geometric margin can be optimised by setting the functional margin $y_i = 1$, with

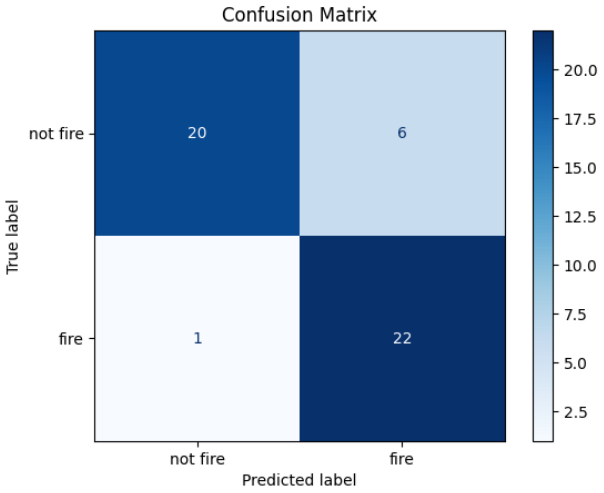

Fig. 2. Confusion matrix for testing set.

$$\langle w \cdot x^+ \rangle + b = 1$$

$$\langle w \cdot x^- \rangle + b = -1$$

This can be represented as inequalities $y_i \left(\langle w \cdot x_i \rangle + b \right) \geq 1 \forall i$.

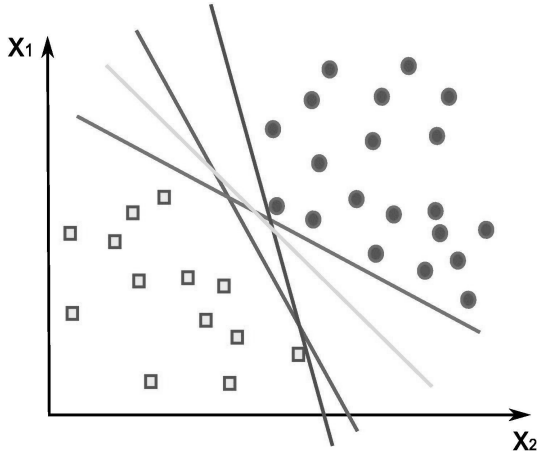

Fig. 3. Decision hyperplanes [7].

In succinct terms, SVM works by finding the hyperplane that best separates data points of different classes in a high-dimensional space [8]. This is done by maximising the margin between the closest points of each class, which are called support vectors. SVM uses kernel functions to handle non-linear classification by mapping data into higher dimensions, where a linear separator can be applied.

The Support Vector Classifier (SVC) model is implemented with a 70/30 data split, allocating 70% for training and 30% for testing. Additionally, a 5-fold cross-validation is conducted to ensure the robustness and reliability of the model's performance. A parameter grid search was performed to find the optimal hyperparameters for the SVC model in the scikit-learn library [9]. The search identified the best parameters as {'C': 100, 'gamma': 0.01, 'kernel': 'poly', 'shrinking': True}. The model was then trained using these parameters, with the predictions and cross-validation results presented in Table 3.

TABLE III
COMPARISON OF EVALUATION METHODS

Method	Training accuracy	Testing accuracy
70/30 split	91%	89%
Cross-validation	92%	85%

The results suggest that the 70/30 split performed better overall compared to the 5-fold cross-validation. Variability in data subsets and randomness in data splitting can lead to varying model performances between a single 70/30 split and a 5-fold cross-validation [10]. The model's classification report is visualised in Table 4.

TABLE IV
CLASSIFICATION REPORT

	precision	recall	f1-score
not fire	100%	75%	86%
fire	84%	100%	91%

The model performs well in distinguishing between Class 1 and Class 0 instances, showing high precision and recall for both categories. It has a precision of 100% for Class 0 and 84% for Class 1, demonstrating high accuracy in its predictions. The model exhibits reliability in its ability to classify instances, particularly in detecting instances of Class 1 with relatively high accuracy.

The model performs well in distinguishing between Class 1 and Class 0 instances, showing high precision and recall for both categories. It has a precision of 100% for Class 0 and 84% for Class 1, demonstrating high accuracy in its predictions. Additionally, it achieves a recall of 100% for Class 1 and 75% for Class 0, effectively capturing instances of Class 1 while moderately identifying instances of Class 0. These results are further supported by F1-scores of 86% for Class 0 and 91% for Class 1, indicating a commendable balance between precision and recall for both classes. The model demonstrates reliability in its ability to classify instances, particularly in detecting instances of Class 1 with high accuracy. These results are confirmed by the confusion matrix presented in Figure 4.

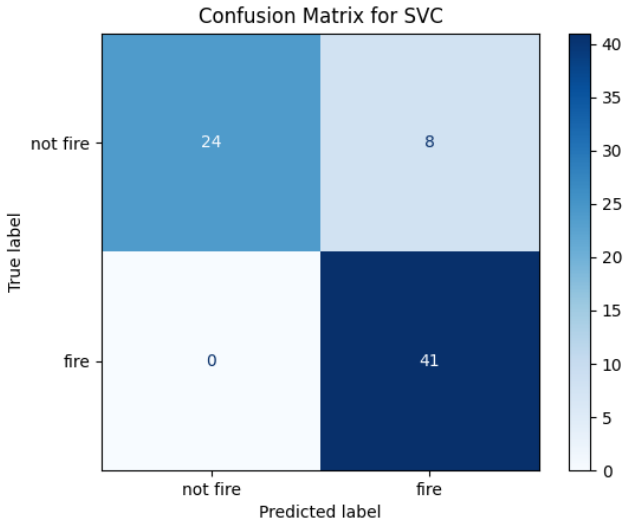

Fig. 4. SVC confusion matrix for testing set.

The confusion matrix shows that the model correctly identified 24 instances of Class 0 (true negatives) while incorrectly classifying 8 instances of Class 0 as Class 1 (false positives). Furthermore, the model accurately detected 41 instances of Class 1 (true positives) and did not produce false negative predictions. These results indicate that the model has a commendable ability to distinguish Class 1 instances, demonstrating a high true positive rate and a minimal false positive rate.

C. K-Nearest Neighbours

K-nearest neighbours (KNN) models operate on the principle that the class label of the closest patterns can

provide valuable information to predict the class label of an unknown pattern x' [3]. This model assigns a class label to a data point by considering the majority class among its K-nearest neighbours in the sample. The measure of similarity in $\mathbb{R}^q$ can be determined using Minkowski's metric expressed by

$$\|x' - x_j\|_p = \left(\sum_{i=1}^{q} |(x_i)' - (x_i)_j|^p \right)^{1/p}$$

which is equal to the Euclidean distance when p = 2. Distance functions, such as the Hamming distance B_q, are essential for different types of data. Binary classification involves problems where the label set is defined as $Y = \{1, -1\}$, and KNN as $f_{KNN}(x') = \begin{cases} 1 & \text{if } \sum_{i \in N_K(x')} y_i \geq 0 \\ -1 & \text{if } \sum_{i \in N_K(x')} y_i < 0 \end{cases}$ with K representing the neighbourhood size and $N_K(x')$ the set of indices of the K-nearest patterns. When $K = 1$, small neighbourhoods emerge in areas where patterns of various classes are distributed. However, when using larger neighbourhood sizes like $K = 20$, patterns with minority labels tend to be disregarded.

In simplified language, a KNN model classifies data points based on the majority class among their nearest neighbours in a feature space, while the value of K determines the number of neighbours considered [11]. This model assumes that similar data points tend to belong to the same class, making it effective for simple classification tasks, but may be sensitive to the choice of K and the distance metric used.

The KNN model was trained using the same 30/70 data split as the SVC model. Implementation of the model followed the guidelines provided in the scikit-learn library [12]. A thorough parameter search was conducted to fine-tune the model's hyperparameters. The optimal parameters {'C': 10, 'gamma': 0.01, 'kernel': 'poly', 'shrinking': True} were identified. Training with these hyperparameters included a 5-fold cross-validation to ensure the accuracy and robustness of the results. The results can be found in Table 5.

The results presented indicate that the 30/70 split and cross-validation performed very close to one another. This means that the model's performance is consistent and not heavily impacted by the random initial data splitting [13]. It indicates that the model can effectively make predictions on new data, giving us confidence in its reliability for future predictions. The model's classification report can be seen in Table 6.

TABLE V
COMPARISON OF EVALUATION METHODS

Method	**Training accuracy**	**Testing accuracy**
70/30 split	84%	74%
Cross-validation	82%	75%

TABLE VI
CLASSIFICATION REPORT

	precision	**recall**	**f1-score**
not fire	70%	63%	67%
fire	76%	81%	79%

The model shows reasonably effective performance in distinguishing between Class 1 and Class 0 instances. It has a precision of 70% for Class 0 and 76% for Class 1, indicating moderate accuracy in its predictions. Furthermore, it achieves a recall of 63% for Class 0 and 81% for Class 1, effectively identifying Class 1 instances while reasonably capturing Class 0 instances. These findings are supported by F1-scores of 67% for Class 0 and 79% for Class 1, indicating a fair balance between precision and recall for both classes. This acquisition is reinforced by the confusion matrix presented in Figure 5.

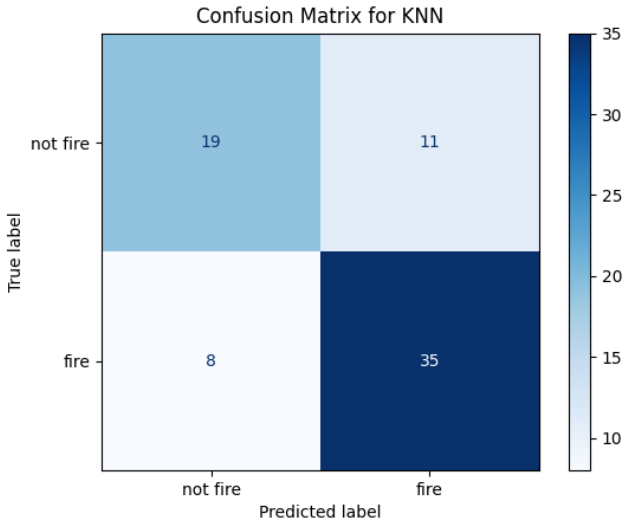

Fig. 5. KNN confusion matrix for testing set.

The confusion matrix shows the performance of a fire prediction model. It correctly identified 19 instances

of Class 0 and 35 instances of Class 1. However, it also made errors: 11 instances of Class 0 were incorrectly predicted as Class 1 (false positives), and 8 instances of Class 1 were incorrectly predicted as Class 0 (false negatives). These results indicate that the model effectively distinguishes fire instances, with a high true positive rate and a relatively low false negative rate. In Section V, the attained results are expounded upon and compared with findings from related studies.

V. Results and Discussion

The study explores how ML can be used for forest fire prediction using meteorological data. This is done using common classification algorithms, including RF, SVM, and KNN, to classify the Algerian Forest Fire data set. This section discusses the results obtained in terms of recall, F1 score, precision, and compares our results with other similar studies. Section A contains a discussion of the results, while Section B elaborates on the analyses and comparison of the findings to related works.

A. Results

The SWM model shows its ability to classify well with the highest scores across all metrics, including a recall of 100%. The RF model ranks second, achieving a recall of 86%, followed by the KNN model, which has a recall of 81%. Compared to the other two models, the KNN scores are relatively low, with a precision of 76%. Although KNN's scores are relatively lower compared to the other two models, it still performed reasonably well. Table VI compares the precision, F1 scores and recall for each model, with the highest values highlighted in bold.

TABLE VII
Comparison of the results of the three models

Models	Precision	Recall	F1-score
RF	79%	96%	86%
SVM	**84**%	**100**%	**91**%
KNN	76%	81%	79%

B. Discussion and Comparison with Related Works

We compared our results with studies that used similar data. Table VIII compares the findings of the two previous studies with accuracy scores. The model in [14] outperforms ours by achieving the highest accuracy score of 93%. Section VI concludes this article by providing a summary of the findings.

TABLE VIII
Comparison of the results of the three models

Ref	Method	Evaluation
Sakr et al. [14]	SVM - ANN	(SVM) accuracy = 93%
Abid, E [15]	DT - RF - adaBoost	(adaBoost) accuracy = 84%
Current study	SVM - RF - KNN	(SVM) accuracy = 89%

VI. Conclusion

Forest fires occur frequently in South Africa, causing substantial damage and threatening lives, property, and the environment. Various ML and DL techniques have been used in literature to detect forest fires or predict their spread, but there is still a need to develop algorithms that can predict a fire occurrence using meteorological data. This study aims to address this by determining whether machine learning (ML) techniques can predict forest fires before they occur using meteorological data. This is done by comparing the results of three ML techniques: Random Forest (RF), Support Vector Machine (SVM), and K-Nearest Neighbours (KNN). Three key evaluation metrics were used to quantify their performance, namely, recall, precision, and F1 score. The SVM algorithms obtained the best results compared to the other algorithms, with a recall score of 100%. The RF model ranked second, achieving a recall of 86%, followed by the KNN model, with a recall of 81%. Compared to the other two models, the KNN scores are relatively low. Lastly, Section VII contains the file list.

VII. File List

- **Dataset:** dataset.csv
- **Code files:** RF Model, SVM & KNN Model.
- **Presentation:** Final presentation.pptx

References

[1] FCSIR, "Wildfires: The impact of climate change on wildfires in south africa.," 2019.

[2] F. Abid, "A survey of machine learning algorithms based forest fires prediction and detection systems," *Fire technology*, vol. 57, no. 2, pp. 559–590, 2021.

[3] O. Kramer, *K-Nearest Neighbors*, pp. 13–23. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[4] F. Abid, "Algerian Forest Fires." UCI Machine Learning Repository, 2019. DOI: https://doi.org/10.24432/C5KW4N.

[5] A. Parmar, R. Katariya, and V. Patel, "A review on random forest: An ensemble classifier," in *International conference on intelligent data communication technologies and internet of things (ICICI) 2018*, pp. 758–763, Springer, 2019.

[6] "Randomforestclassifier." https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. Accessed: 21 May 2024.

[7] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez, "A comprehensive survey on support vector machine classification: Applications, challenges and trends," *Neurocomputing*, vol. 408, pp. 189–215, 2020.

[8] M. Awad and R. Khanna, *Support Vector Machines for Classification*, pp. 39–66. Berkeley, CA: Apress, 2015.

[9] "Svc." https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html. Accessed: 21 May 2024.

[10] T. H. Stephen Bates and R. Tibshirani, "Cross-validation: What does it estimate and how well does it do it?," *Journal of the American Statistical Association*, pp. 1–12, 2023.

[11] Z. Zhang, "Introduction to machine learning: k-nearest neighbors," *Ann Transl Med*, vol. 4, p. 218, Jun 2016.

[12] "Kneighborsclassifier." https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.htmlkneighborsclassifier. Accessed: 21 May 2024.

[13] S. Yadav and S. Shukla, "Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification," in *2016 IEEE 6th International Conference on Advanced Computing (IACC)*, pp. 78–83, 2016.

[14] G. E. Sakr, I. H. Elhajj, and G. Mitri, "Efficient forest fire occurrence prediction for developing countries using two weather parameters," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 5, pp. 888–894, 2011.

[15] F. Abid and N. Izeboudjen, "Predicting forest fire in algeria using data mining techniques: Case study of the decision tree algorithm," in *Advanced Intelligent Systems for Sustainable Development (AI2SD'2019) Volume 4-Advanced Intelligent Systems for Applied Computing Sciences*, pp. 363–370, Springer, 2020.