applied sciences

MDPI

Article

Hierarchical Fusion of Machine Learning Algorithms in Indoor Positioning and Localization

Ahmet Çağdaş Seçkin [1,*] and Aysun Coşkun [2]

[1] Department of Mechatronics, Vocational School of Technical Sciences, Uşak University, Uşak 64100, Turkey
[2] Department of Computer Engineering, Faculty of Technology, Gazi University, Ankara 06100, Turkey
* Correspondence: cagdas.seckin@usak.edu.tr

Received: 8 August 2019; Accepted: 1 September 2019; Published: 4 September 2019

Abstract: Wi-Fi-based indoor positioning offers significant opportunities for numerous applications. Examining the Wi-Fi positioning systems, it was observed that hundreds of variables were used even when variable reduction was applied. This reveals a structure that is difficult to repeat and is far from producing a common solution for real-life applications. It aims to create a common and standardized dataset for indoor positioning and localization and present a system that can perform estimations using this dataset. To that end, machine learning (ML) methods are compared and the results of successful methods with hierarchical inclusion are then investigated. Further, new features are generated according to the measurement point obtained from the dataset. Subsequently, learning models are selected according to the performance metrics for the estimation of location and position. These learning models are then fused hierarchically using deductive reasoning. Using the proposed method, estimation of location and position has proved to be more successful by using fewer variables than the current studies. This paper, thus, identifies a lack of applicability present in the research community and solves it using the proposed method. It suggests that the proposed method results in a significant improvement for the estimation of floor and longitude.

Keywords: hierarchical data; fingerprinting; indoor positioning; information fusion; localization; machine learning; Wi-Fi

1. Introduction

Today, determination of indoor location and position of objects is an important subject, which provides many applications in various fields such as robotics, asset tracking, warehouse management, crowd analysis [1–4]. The idea of indoor positioning and localization based on the strength of wireless signal was first introduced in 2000 [5]. Since then, it has sparked great interest and become the subject of many research areas. The concepts of positioning and localization are often confused. The position of an object corresponds to a specific point in a coordinate system such as longitude and latitude. However, the location of an object provides details of the position such as office, floor, building, etc. In this regard, localization can be defined as a simpler or categorical state of the positioning process. Localization is sufficient for applications that require less precision such as crowd analysis and storage-asset tracking. Positioning, on the other hand, is required for more precise operations such as robotics.

Today, creating an economic, accurate, and precise system by utilizing the already installed Wi-Fi infrastructure is one of the most important goals of the research community. To this end, it is essential to use the pattern created by the propagation and address broadcasting of the Wi-Fi access points. The pattern provided by the numerous access points that are fixed to various points becomes unique by undergoing various reflections and weaknesses with the contribution of its spatial structure. This unique pattern is called Wi-Fi fingerprint. The simple Wi-Fi fingerprint data consist of the received signal strength

indicator (RSSI) and the media access control (MAC) address data. RSSI is a term used to measure the quality of a received signal of a client device. RSSI acts as a relative received signal strength because each chip manufacturer has its own receiver characteristics for the standard of IEEE 802.11. RSSI is simply an indication of the power level received by the receiver after the antenna and possible cable losses. Thus, the higher the RSSI value is, the stronger the signal is. If two RSSI values are represented as negative (for example, −100 and −50), the one that is closer to 0 (−50) is the stronger received signal. However, when the RSSI information is analyzed with the spectrum analyzer, new fingerprint data called channel status information (CSI) are generated. In wireless communications, CSI refers to known channel properties of a communication link. It describes how a signal propagates from the transmitter to the receiver and represents the combined effect of scattering, fading, and power decay with distance. Short-term CSI describes instant or very short listening/viewing of an existing channel condition. Long-term CSI, on the other hand, describes a statistical characterization of the data obtained as a result of the channel being listened to for long enough to eliminate the influence of other systems in the environment and periodical factors such as overtime, season of the year, or other patterns. In CSI, 14 new features are included in the analysis only when the channel power information is used because there are 14 main channels in IEEE 802.11. Further, the obtained data are mainly affected by environmental effects and channel usage; therefore, time series analysis is required. It has been demonstrated in many studies that successful positioning operations are performed by considering these points [6–9]. While examining the positioning operations with CSI, it is important to note that data coming from the spectrum analyzer have been used in the positioning process to achieve high performance. However, this feature is not supported by mobile devices because they are not fully applicable for it.

Until today, many positioning systems based on the changes in the RSSI indicator obtained from Wi-Fi or Bluetooth signals have been developed [10–14]. Moreover, initiatives inspired by these studies such as Insiteo [15], Wifarer [16], Insoft [17], The Framework for Internal Navigation and Discovery (FIND) [18], and MapsIndoors [19] were launched as well. Successful studies with Wi-Fi fingerprint positioning are found in various areas of subjects today [12,20]. One of the most recent datasets for Wi-Fi fingerprint-based localization is named "UJIIndoorLoc." This dataset is obtained by Torres-Sospedra et al. and makes use of diverse types of studies [12,21,22]. These studies are mainly focused on adopting the same 525 features. In the process of classification and regression, it is challenging, time-consuming, and inefficient to apply all the input values directly to learning algorithms. One of the preferred methods in data reduction is principal component analysis (PCA) [23]. Some researchers use PCA and/or linear discriminant analysis (LDA) to reduce the data features [24]. The total number of features for these studies is greater than 100, even when sufficient raw features are covered by the reduced feature set. Since the number of wireless access points (WAP) may vary according to the environment of application and too many features may be employed, it is difficult and inefficient to implement it for real-world indoor positioning. Using the PCA method, new components are created according to the feature set, and it is possible to cover all of the features. However, the PCA method is specifically designed for the structure of that dataset. In this regard, if the Wi-Fi infrastructure is changed, the PCA-generated system will become useless because of the need to change the properties of its components. The dataset that is to be used in real-world applications should have common features found in every environment while also being compatible with the changing number of WAP and containing few features.

Investigating the previous studies, it was observed that a large number of features were used while the existing dataset was used without adequate simplification. Prominent studies were also found in the literature; however, these were either not practical or only valid for specific cases. Thus, it was not possible to provide an application with common parameters found in every environment. The main purpose of this study is to develop a method that can be applied for any environment where WAP mapping data are collected. It aims to create a common and standardized dataset for indoor positioning and localization and present a system that can perform estimations using this dataset. In order to achieve this goal, machine learning (ML) methods are compared and the results of successful

ML methods with hierarchical inclusion are then investigated. Further, new features are generated according to the measurement point obtained from "UJIIndoorLoc" dataset. Subsequently, learning models are selected according to the performance metrics for the estimation of location and position. These learning models are then fused hierarchically using deductive reasoning. The second part of the paper introduces the dataset adopted in this study and the method of positioning. The third part presents the results of the study. Finally, a conclusion is suggested according to the results. The main contribution of this study will be to propose a more general dataset structure for datasets such as localization and positioning in the hierarchical structure and to provide a more successful retrofit method in hierarchical situations.

2. Materials and Methods

The overall system for Wi-Fi-based positioning and localization consists of fixed WAP's, mobile devices, and positioning services. The stages of this general method are shown in Figure 1. To start with, new features are generated from the dataset. The generated dataset is split into two categories, namely training and test. This process is carried out randomly and the ratio of training data and test data is 70% and 30%, respectively. In the next stage, training data are presented to Gaussian naive Bayes (NB), decision tree (DT), K-nearest neighbors (KNN), support vector machine (SVM), random forest (RF), and multi-layer perceptron (MLP) algorithms. These learning algorithms were obtained from Scikit-learn library [25–29]. They are then compared according to the classification and regression performance metrics, which are explained in the ML section, while the most suitable algorithms are selected for positioning and localization. In addition, the hierarchical fusing ML (HF-ML) algorithm is shown in Figure 2. In this algorithm, the dataset is primarily used for the training of building classification. Results of the building prediction are then combined with the input dataset. Consequently, prediction results and the input dataset are used for the training of floor prediction. After the training, floor predictions are combined with the previous dataset. This dataset contains the input dataset, building predictions, and floor predictions. The resulting dataset of these combined hierarchical prediction procedures is used in the training of longitude and latitude regression. The selected learning models are fused hierarchically using deductive reasoning. According to HF-ML, there are two more types of input data, namely building information and floor information. Since two new variables were introduced, HF-ML was trained with the training data and the performance was evaluated with the test data. This obtained model can produce location and position data after applying the feature generation process to the new data. See the supplementary materials for the codes and the data.

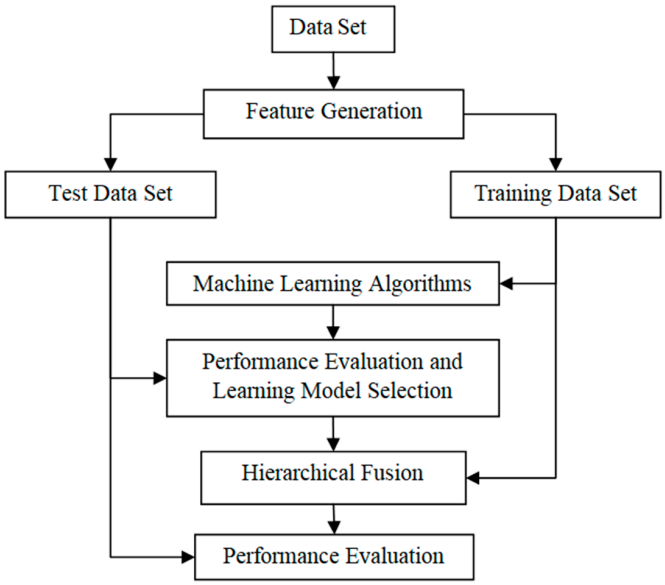

Figure 1. Positioning and localization diagram.

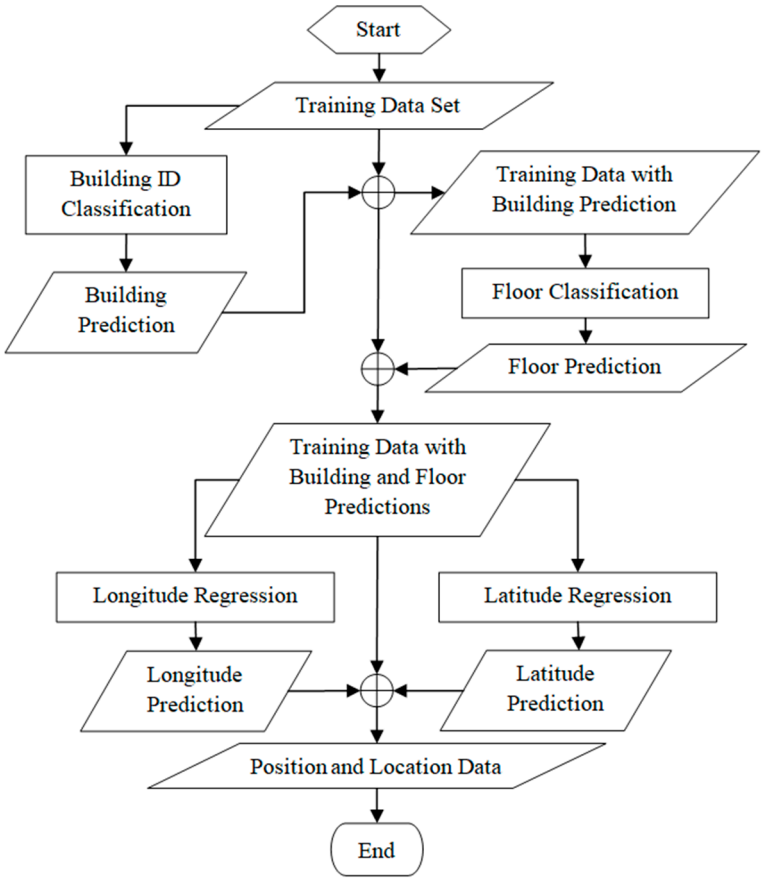

Figure 2. Hierarchical fusion algorithm.

2.1. Raw Dataset and Properties

This study uses the dataset named "UJIIndoorLoc", which was obtained at Jaume I University [21]. The dataset was collected from 3 different buildings, each of them having 5 floors. A total of more than 20,946 samples were collected. This dataset is publicly available online [30], and the variables in it are explained in Table 1. features, and their range are outlined in this table below.

Table 1. Dataset properties.

Feature	Properties	Range
1–520	WAP RSSI level	from −104 to 0 and +100
521	Longitude	from −7695.9387549299299000 to −7299.786516730871000
522	Latitude	from 4,864,745.7450159714 to 4,865,017.3646842018.
523	Floor	Integer values from 0 to 4
524	Building ID	Categorical integer values from 0 to 2
525	Space ID	Where the capture was taken. Categorical integer values (office, corridor, classroom)
526	Relative Position	1—Inside, 2—Outside, in front of the door
527	User ID	Categorical integer values from 0 to 18
528	Phone ID	Categorical integer values from 0 to 18
529	Timestamp	UNIX Time

WAP: Wireless Access Point; RSSI: Received Signal Strength Indicator; ID: Identifier.

2.2. Feature Generation

For the method employed in the study, no data based on the user id, timestamp, phone id, and space id were used. The study is based on the production of a new dataset in order to provide an alternative to a system that is less applicable due to the dependence of the data on a specific location, user, phone, or WAP number. Also, the generated features are explained in Table 2. In this table, abbreviations of generated features are presented according to the data type and each of these features are explained. Moreover, the absolute Pearson cross-correlation coefficients of the generated data are shown in Figure 3. According to this figure, building, longitude, and latitude values are highly correlated. Also, FirstMAC, SecondMAC, and ThirdMAC features provide better correlation for output values.

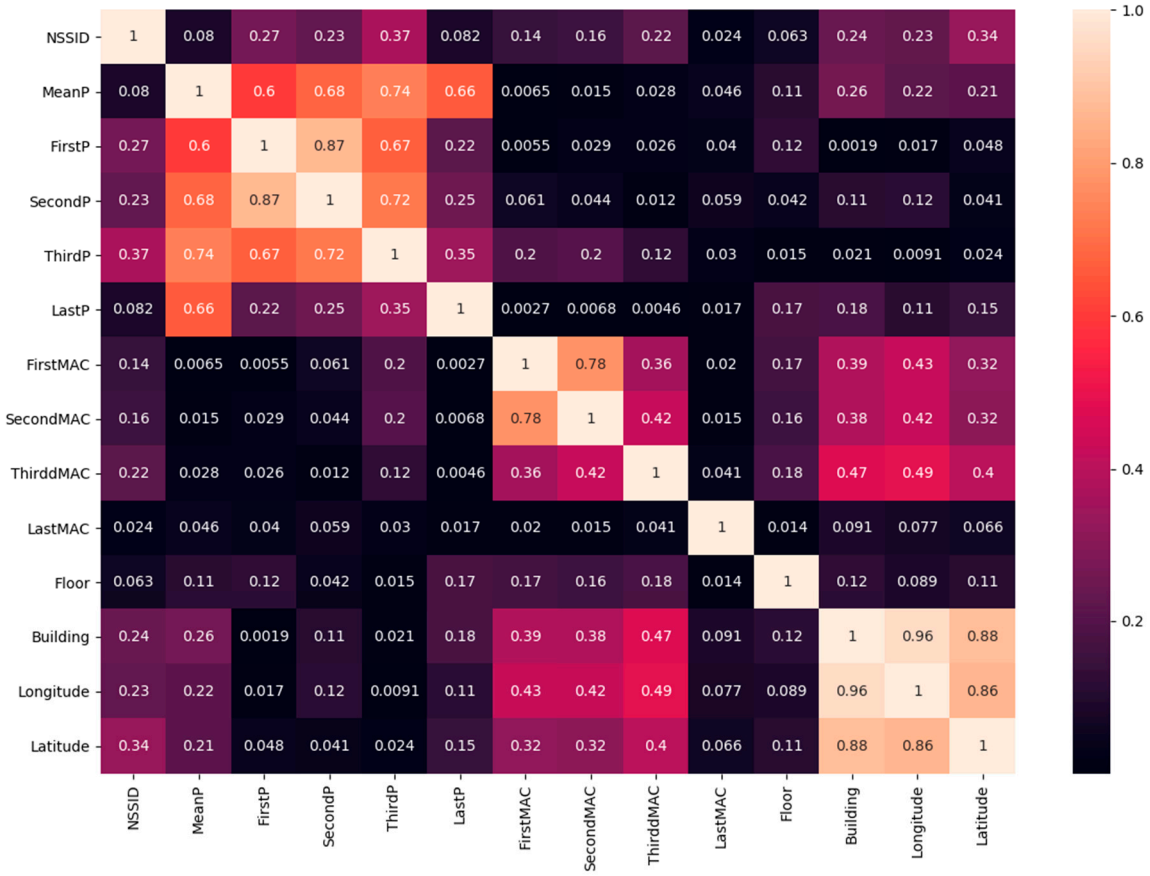

Figure 3. Absolute cross-correlation coefficients of dataset.

Table 2. Generated features.

Feature Abbreviation	Type	Explanation
FirstMAC	Categorical	WAP ID of the highest RSSI value.
SecondMAC	Categorical	WAP ID of the second-highest RSSI value. If not, use the First WAP ID.
ThirdMAC	Categorical	WAP ID of the third-highest RSSI value. If not, use the Second WAP ID.
LastMAC	Categorical	WAP ID of the lowest RSSI value. If not, use the Third WAP ID.
FirstP	Numeric	RSSI value of the first WAP.
SecondP	Numeric	RSSI value of the second WAP. If not, use the First WAP RSSI.
ThirdP	Numeric	RSSI value of the third WAP. If not, use the Second WAP RSSI.
LastP	Numeric	RSSI value of the last WAP. If not, use the Third WAP RSSI.
NSSID	Numeric	Number of WAP with a RSSI level between 0 and −104. Value of +100 is excluded because it means there is no signal according to the dataset.
MeanP	Numeric	Mean RSSI values of WAP with a RSSI level between 0 and −104

2.3. Machine Learning

Gaussian naive Bayes (GNB) classifier is a probabilistic classifier algorithm based on Bayes' theorem [31]. Due to its low computational load and high correct classification performance, it has been widely chosen in the studies aimed for classification. The aim of the naive Bayes classifier is to compute the i-th observation that is obtained by computing the posterior probability of class c for a given feature x_i. Calculation of the posterior probability is given in Equation (1). In this equation $P(c|x_i)$, is the posterior probability while $P(c)$ is class prior probability and $P(x_i|c)$ is the likelihood, which means probability of i-th feature, and finally, $P(x_i)$ is the prior probability of feature i. There are three main steps in the naive Bayes algorithm. In the first step, the algorithm calculates the frequency values of each output category according to the dataset. In the second step, the likelihood table is calculated. In the last step, probability of output class according to Gaussian Distribution is calculated [31,32].

$$P(c|x_i) = \frac{P(c)P(x_i|c)}{P(x_i)} \quad (1)$$

The K nearest neighbors (KNN) algorithm is a learning algorithm that operates according to the values of the nearest k-neighbor. The KNN algorithm is a non-parametric method for classification and regression [33]. It was first applied to the classification of news articles [34]. When performing learning with the KNN algorithm, firstly, the distance of the data to others is calculated in the dataset examined. This length calculation is done with Euclidian, Manhattan, or Hamming distance function. Then, the mean value of the nearest K neighbors is calculated for the data. The K value is the only hyper-parameter of the KNN algorithm. If the K value is too low, then the borders are going to be flickering and overfitting will occur, whereas if the K value is too high, the separation borders are going to be smoother and underfitting will occur. The disadvantage of the KNN algorithm is that in the distance calculation process, it increases the processing load as the number of data increases.

Decision tree (DT) algorithm is an algorithm that is frequently used in statistical learning and data mining. It works with simple if-then-else decision rules and is used for both classification and regression in supervised learning [35]. It has three basic steps. In the first step, the most meaningful feature is placed as the first (root) node. In the second step, datasets are divided into subsets according to this node. Subsets should be created in such a way that each subset contains data with the same value of a feature. In the third step, step one and two are repeated until the last (leaf) nodes in all of the branches are created. The decision tree algorithm builds classification or regression models in the form of a tree structure. It splits a dataset into smaller subsets while at the same time an associated decision tree is incrementally developed. The result of this algorithm is a tree with decision nodes. Decision trees can handle both categorical and numerical data [36,37].

The RF method is designed in the form of a forest consisting of many DT's [38]. Each decision tree in the forest is formed by selecting the sample from the original dataset with the bootstrap technique and selecting the random number of all variables in each decision node. The RF algorithm consists of four fundamental steps. First, n features are randomly selected from a total of m features. Second, the node d using the best split point is calculated among the n features. Third, it checks whether the number of final (leaf) nodes reaches the target number. If it does, it moves on with the next step; if not, the algorithm goes back to step one. Finally, by repeating steps one to three for n (number of trees in the forest) times, a forest is built [38–40].

The adaptive boosting (AB) algorithm is a ML method, which is also known as an ensemble method. The aim of this method is to create a stronger learning structure by using weak learners. AB can be used to improve the performance of any ML algorithm. However, it often uses a single-level decision tree algorithm as a weak learner algorithm since the processing load is much lower than other basic learning algorithms. The AB algorithm consists of four basic steps. In the first step, the N weak algorithms are run and the dataset is learned. Each of these N weak learning algorithms is assigned a weight value of 1/N. In the second step, error values are calculated in each of the learner algorithms.

In the third step, the weight value of the learning algorithm with a high amount of error is increased. In the final step, the learning algorithms are summed with weight values and if the desired metric limit is reached, the total algorithm is output; otherwise, it returns to the second step [41–43]. As the number of learners in this algorithm increases, the processing load and the learning performance increases.

First introduced by Vapnik, SVM is a supervised learning algorithm based on the statistical learning theory [44,45]. SVM was originally developed for binary classification problems. While SVM was only used for classification at first, it eventually began to be used in linear and non-linear regression procedures as well [46]. SVM aims to find the hyper-plane, which separates classes from each other, and which is the most distant from both classes. In cases where a linear separation cannot occur, the data are moved to another space of a higher dimension, and the classification is performed in that space. While the classification processes aim to separate data from each other by means of the generated vectors; the regression, in a sense, performs the opposite of this process and aims to create a hyper-plane by identifying support vectors in a way that they will include as many data as possible [47]. The SVM is mainly divided into two categories according to the linear separability of the dataset. Non-linear SVM decision function is given in Equation (2). In cases where the data cannot be separated linearly, the data are moved to a space of higher dimension, and kernel functions are used to resolve them. The transformations can be made by using Kernel functions expressed as $K(x_i,x_j) = \Phi(x)\ \Phi(x_i)$ instead of $\Phi(x)\ \Phi(x_i)$ scalar product given in Equation (2). Also, non-linear transformations can be made, and the data can be separated in the high dimension thanks to the Kernel functions. Therefore, Kernel functions have a critical role in the performance of SVM. The most widely used of these Kernel functions are linear, polynomial, Radial Basis Function (RBF), and Sigmoid, all of which are given in Table 3 [47–49]. In this table, x_i and x_j corresponds to window coefficients. x_i is the width parameter and the x_j represents the place of the window. In the polynomial function, the parameter d corresponds to the degree of the polynomial. In radial basis function, on the other hand, γ is equal to Gaussian function. In this study, only RBF was used as the SVM Kernel function.

$$f(x) = \text{sign}(\sum_{i=1}^{N} a_i y_i \Phi(x)\Phi(x_i) + b) \quad (2)$$

Table 3. Support vector machine (SVM) kernel functions.

Kernel Function	Equation
Linear	$K(x_i, x_j) = (x_i^T x_j)$
Polynomial	$K(x_i, x_j) = (x_i x_j)^d$
Radial Basis Function (RBF)	$K(x_i, x_j) = \exp(-\gamma\|x_i - x_j\|^2), \gamma > 0$
Sigmoid	$K(x_i, x_j) = \tanh(kx_i x_j - \delta)$

K-fold cross validation is a method used in the performance evaluation of learning algorithms [50,51]. In the evaluation processes, it is essential to create a dataset for training and testing and to test the performance of the model that actualizes the learning with training data on the test dataset. However, performance evaluation may not be reliable because of not having the same distribution when selecting the training and test data in the dataset, uneven distribution of outliers, and so on. Therefore, K-fold cross validation method was developed. In this method, training and test data are integrated and turned into a single dataset. All data are divided into K equal parts. The K value here is determined by the user. Then, learning and testing are performed for each of the K sub-sets; here, one of the subsets will be used for the test, and the others are going to be used for the training. As a result, performance metrics are obtained for each of the sub-sets. The average of the performance metrics is considered as the performance metric of the K-fold cross-validation. In this study, the K value was chosen as 10.

The metrics of classification performance are obtained through the confusion matrix in Figure 4 as outlined in Table 4. The true positive (TP) value in a two-class confusion matrix is the number of predictions where the predicted value is 1 (true) when the actual value is also 1 (true). The true negative (TN) value is the number of predictions where the predicted value is 0 (false) when the actual value is also 0 (false). The false positive (FP) value is the number of predictions where the predicted value is 1 (true) when the actual value is 0 (false). The false negative (FN) value is the number of predictions where the predicted value is 0 (false) when the actual value is 1 (true). The mean square error (MSE), mean average error (MAE), and coefficient of determination (R^2) metrics used in the regression performance evaluation are outlined in Table 5. It represents the sample standard deviation of the differences between predicted values ($\hat{y}_j$) and observed values (y_j).

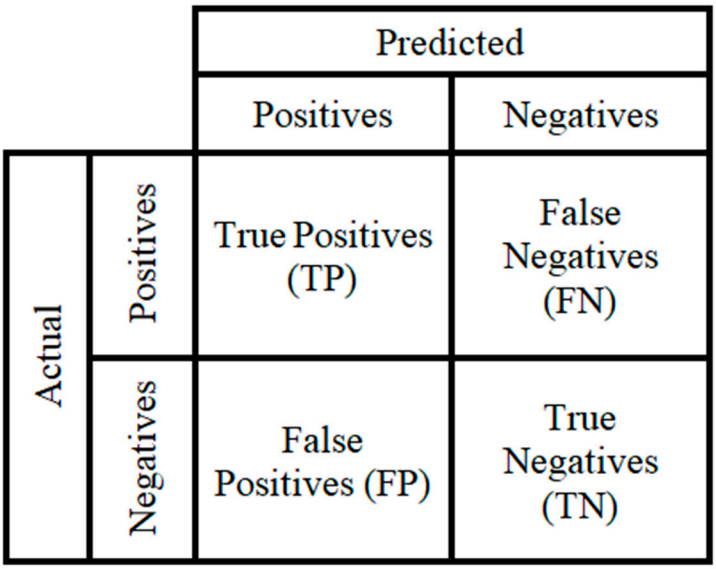

Figure 4. Confusion matrix.

Table 4. Classification metrics.

Metric	Equation
Accuracy	$A = \frac{TN+TP}{TP+TN+FP+FN}$
Precision	$P = \frac{TP}{TP+FP}$
Recall	$R = \frac{TP}{TP+FN}$
F1 Score	$F1 = \frac{P+R}{2}$

Table 5. Regression metrics.

Metric	Equation
Mean Square Error (MSE)	$MSE = \frac{1}{n}\sum_{j=1}^{n}\left(y_j - \hat{y}_j\right)^2$
Mean Average Error (MAE)	$MAE = \frac{1}{n}\sum_{j=1}^{n}\left\|y_j - \hat{y}_j\right\|$
R Squared (R^2)	$R^2 = 1 - \frac{\sum_{j=1}^{n}\left(y_j-\hat{y}_j\right)^2}{\sum_{j=1}^{n}\left(y_j-\bar{y}\right)^2}$

3. Results and Discussion

In this section, HF-ML localization and positioning results are presented. The generated dataset consists of 20,946 rows with 10 columns of constructed features. When the data are split (70% training, 30% test), 4190 samples are available for test and 16,756 for training.

3.1. Building Classification Results

Building classification is the first step in HF-ML localization and positioning. Only the generated data are used for classification. The results obtained by ML methods are presented in Table 6. The accuracy value in this table is provided for both the training and testing. The value given for the training is the average of the results obtained in the 10-fold cross-validation stage. The precision, recall, and F1 values obtained for the testing are calculated according to the distribution weights, since there are three different output values and these outputs are in different numbers in the dataset. According to the findings, the RF algorithm is considered to be the most successful algorithm in both training and test metrics. When the results are examined in terms of accuracy metrics, the RF algorithm produces a value of 0.993 in training and a value of 0.995 in the test. According to the results of the RF algorithm for the testing data, the confusion matrix is presented in Figure 5. In this figure, it is seen that the actual values and the predicted values match to a very high degree. Also, there are small numbers of confusion between close buildings.

Table 6. Building classification results.

ML Method	Accuracy		Precision	Recall	F1
	Training	Testing	Testing	Testing	Testing
NB	0.673	0.670	0.722	0.670	0.674
KNN	0.963	0.964	0.964	0.964	0.964
DT	0.987	0.993	0.993	0.993	0.993
RF	0.993	0.995	0.995	0.995	0.995
AB	0.916	0.897	0.896	0.897	0.896
SVM	0.605	0.620	0.790	0.620	0.569

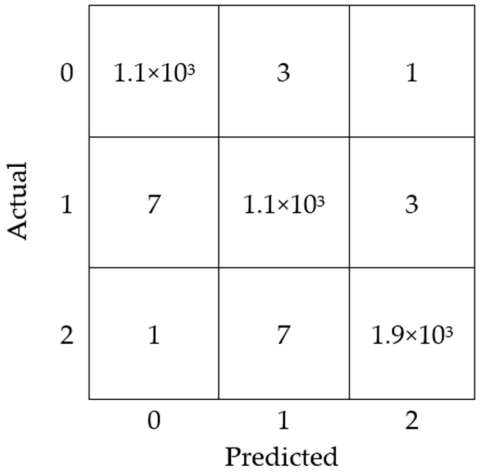

Figure 5. Confusion matrix of random forest (RF) building classification.

3.2. Floor Classification Results

Floor classification is the second step in HF-ML localization and positioning. For the classification, the generated data and the predictions produced by the selected RF algorithm according to the results presented in the previous section are used. The results obtained with the ML methods are presented in Table 7. The accuracy value in this table is provided for both training and testing. The value given for the training is the average of the results obtained in the 10-fold cross-validation stage. The precision, recall, and F1 values obtained for the testing are calculated according to the distribution weights since there are five different output values and these outputs are in different numbers in the dataset. The HF tag is added when the generated data are combined with the predictions of RF building classification. In general, when the performance metrics are considered, the positive effect of HF is

observed. According to the findings, the HF-RF algorithm is considered to be the most successful one in terms of both the training and test values. In terms of accuracy metrics, the HF-RF algorithm produces a high score of 0.951 in training and 0.960 in testing. The confusion matrix of the RF and HF-RF algorithms are presented in Figures 6 and 7, respectively. In these figures, it is seen that the actual and the predicted values match to a very high degree. Also, there are small numbers of confusion between floors close to each other. Comparing Figures 6 and 7, the decrease in the wrong estimated values is clearly seen.

Table 7. Floor classification results.

ML Method	Accuracy		Precision	Recall	F1
	Training	Testing	Testing	Testing	Testing
NB	0.340	0.351	0.354	0.351	0.329
KNN	0.810	0.819	0.820	0.819	0.818
DT	0.940	0.953	0.953	0.953	0.953
RF	0.947	0.956	0.957	0.956	0.956
AB	0.580	0.598	0.599	0.598	0.595
SVM	0.552	0.571	0.832	0.571	0.577
HF-NB	0.277	0.283	0.391	0.283	0.279
HF-KNN	0.810	0.819	0.820	0.819	0.819
HF-DT	0.942	0.950	0.950	0.950	0.950
HF-RF	0.951	0.960	0.960	0.960	0.960
HF-AB	0.473	0.508	0.542	0.508	0.512
HF-SVM	0.563	0.585	0.834	0.585	0.594

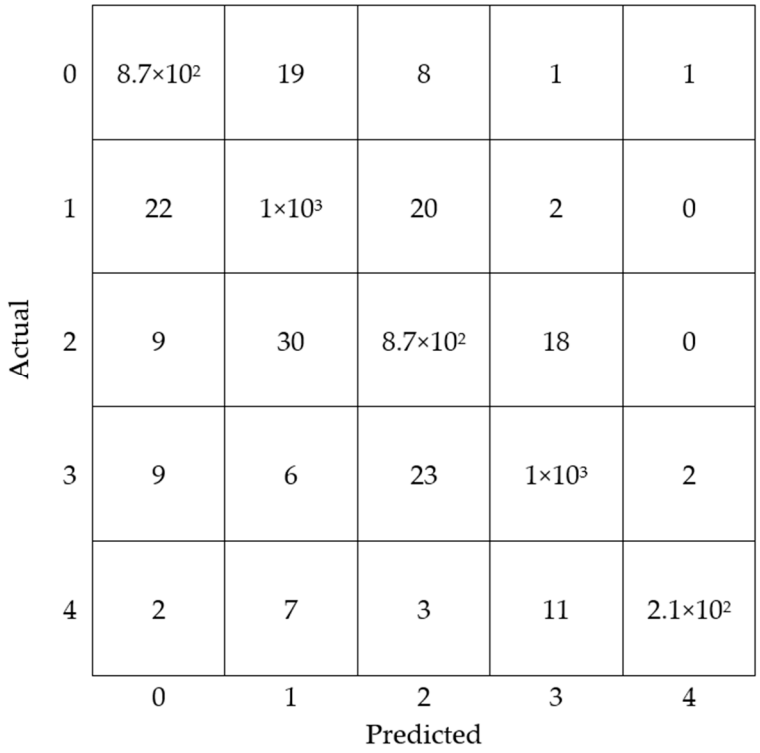

Figure 6. Confusion matrix of RF floor classification.

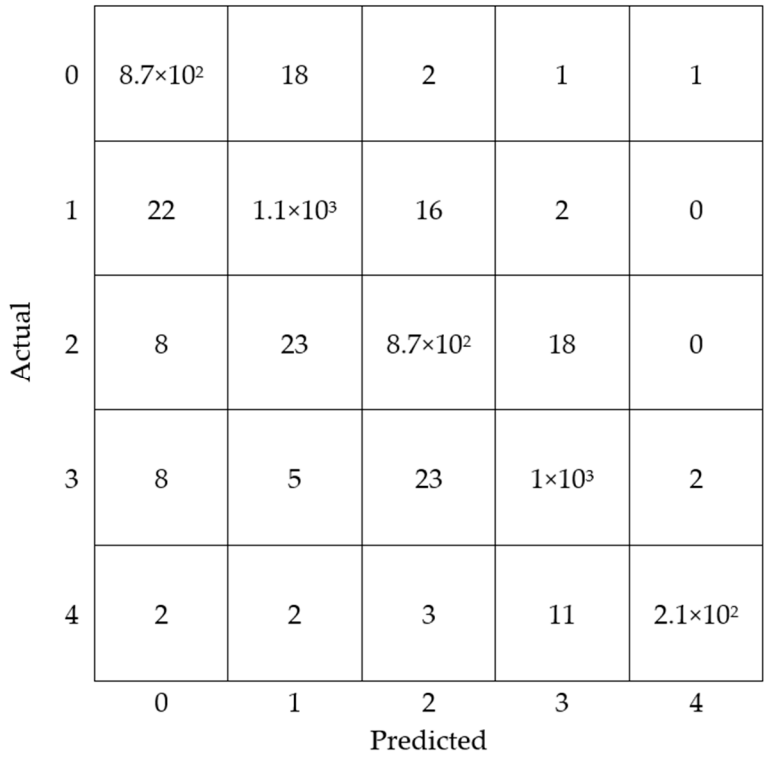

Figure 7. Confusion matrix of hierarchical fusing (HF)-RF floor classification.

3.3. Longitude and Latitude Regression

The algorithms for longitude and latitude estimation have been analyzed with ML using the generated data and additional HF data. Previously, RF was used for building classification and HF-RF for floor classification. The NB algorithm was excluded from the longitude and latitude estimation because it was not used in the regression process. Performance metrics of longitude regression are presented in Tables 8 and 9. Considering the general performance metrics, application of HF has a positive effect on longitude estimation. When the algorithms are analyzed in terms of MSE metrics, the most successful algorithm turns out to be the HF-DT. While the MSE value of the regression with DT was 270.975, the MSE value of HF-DT decreased to 246.582. The results obtained with HF-DT for training data have a very low MSE value of 3.751.

Table 8. Longitude regression results of training data.

ML Method	MSE	MAE	R^2
KNN	764.771	12.602	0.951
DT	3.751	0.287	1.000
RF	37.373	2.552	0.998
AB	4596.002	59.143	0.703
SVM	16,669.187	106.174	−0.077
HF-KNN	755.945	12.510	0.951
HF-DT	3.751	0.287	1.000
HF-RF	17.743	2.127	0.999
HF-AB	977.574	25.440	0.937
HF-SVM	16,655.674	106.123	−0.076

Table 9. Longitude regression results of testing data.

ML Method	MSE	MAE	R^2
KNN	1306.443	16.713	0.914
DT	270.975	5.495	0.982
RF	173.116	5.939	0.989
AB	4599.668	59.047	0.696
SVM	16,257.159	105.057	−0.076
HF-KNN	1284.751	16.602	0.915
HF-DT	246.582	5.596	0.984
HF-RF	201.812	5.646	0.987
HF-AB	1089.982	26.149	0.928
HF-SVM	16,243.445	105.003	−0.075

The performance metrics obtained as a result of latitude regression are presented in Tables 10 and 11. Considering the general performance metrics, application of HF has a negative effect on latitude estimation. When the algorithms are examined for MSE metrics, DT appears to be the most successful algorithm. While the MSE value of the regression with DT is 190.452, the MSE value of HF-DT is 1284.751. The results obtained with DT for training data have a very low MSE value of 2.171.

Table 10. Latitude regression results of training data.

ML Method	MSE	MAE	R^2
KNN	326.695	9.212	0.928
DT	2.171	0.212	1.000
RF	19.486	2.064	0.996
AB	1571.732	32.841	0.655
SVM	4567.286	54.533	−0.002
HF-KNN	755.945	12.510	0.951
HF-DT	3.751	0.287	1.000
HF-RF	17.743	2.127	0.999
HF-AB	977.574	25.440	0.937
HF-SVM	16,655.674	106.123	−0.076

Table 11. Latitude regression results of testing data.

ML Method	MSE	MAE	R^2
KNN	550.830	12.227	0.877
DT	190.452	5.107	0.957
RF	98.939	4.924	0.978
AB	1573.053	32.894	0.648
SVM	4538.017	54.721	−0.015
HF-KNN	1284.751	16.602	0.915
HF-DT	246.582	5.596	0.984
HF-RF	201.812	5.646	0.987
HF-AB	1089.982	26.149	0.928
HF-SVM	16,243.445	105.003	−0.075

3.4. Comparison of Findings with Other Studies

Comparison of localization studies are presented in Table 12 and the comparison of positioning studies are given in Table 13. According to the Table 12, the most successful method seems to be the one proposed by Bozkurt et.al. [12]. However, when the average of accuracy values obtained as a result of the proposed method is considered, it ranks second among these studies. According to the Table 13, the proposed method HF-DT appears to be the most successful one. The results of the extra-trees are given as root mean square error; however, to compare with the values present in this study, the square

of the given values are converted into the MSE metrics [11]. Carrying out a literature review, it was found that the only study to perform both localization and positioning procedures was conducted by Akram et.al. Also, according to the findings, it was seen that the proposed method provides higher performance values [14].

Table 12. Comparison of localization studies.

Method [Citiation]	Building Accuracy	Floor Accuracy	Mean
[12]	0.99	0.98	0.985
[52]	0.98	0.92	0.95
HybLoc [14]	0.85	0.85	0.85
Proposed Building Model (RF)	0.99	-	0.97
Proposed Floor Model (HF-RF)	-	0.95	

Table 13. Comparison of positioning studies.

Method [Citiation]	Longitude MSE	Latitude MSE	Longitude MAE	Latitude MAE
KNN [11]	205.636	180.365	-	-
SVM [11]	252.81	127.238	-	-
Extra-Trees [11]	149.084	102.414	-	-
HybLoc [14]			6.46	6.46
Proposed Longitude Method (HF-DT)	3.751	-	0.287	0.212
Proposed Method (DT)	-	2.171	-	-

4. Conclusions

In this study, first of all, a more applicable, standard, and simple dataset structure was designed, and then a hierarchical structure of the new dataset as well as a higher performance method was introduced. Using the proposed method, estimation of location and position has proved to be more successful by using fewer variables than the current studies. This paper, thus, identifies a lack of applicability present in the research community and solves it using the proposed method. According to the results obtained with the application of the method, the following conclusions were obtained. A simple, standardized, and more efficient dataset has been proposed instead of an indoor positioning dataset whose implementation is difficult and inefficient. This new, simple, and more standardized dataset that was proposed instead of the classic dataset has achieved high performance values. Also, a hierarchical fusion method was proposed to improve the classical ML models. The proposed HF method resulted in a significant improvement in floor and longitude estimation. The method was developed from the general to the specific using the Hierarchical structure of the localization process. Here, by transferring the estimation, it is ensured that the correct parts of the algorithms are strengthened.

When the findings were examined, RF was found to be the most successful algorithm for building classification. In terms of accuracy metrics, the RF algorithm produces a value of 0.993 in training and a value of 0.995 in testing. Also, HF-RF is chosen as the most successful method for floor classification. In terms of accuracy metrics, the HF-RF algorithm produces a high score of 0.951 in training and 0.960 in testing. In terms of localization, this study concludes that the variable production and the HF method provide efficiency.

The HF method has led to an improvement in the longitude estimation process, while it has also been shown to cause deterioration in the latitude estimation process. For this reason, in the estimation of longitude, HF-DT, which has a 3.751 MSE, was preferred and DT, which has a 3.751 MSE, was preferred in latitude estimation. In terms of positioning, this study concludes that the variable production and the HF method provide efficiency. Additionally, it was observed that the method of variable production proves to be successful. Further, it was concluded that the HF method only

yields longitude estimation. Therefore, it is promising for the position estimation in indoor areas such as rooms, corridors, etc. Although there was no attempt to estimate the coordinates in any of the similar studies, a coordinate estimation was carried out in this study. However, the coordinate estimate was not as precise as GPS. This study serves as a preliminary study for future localization or indoor localization systems using Wi-Fi. Thanks to the results of this study, a strengthening technique with HF-ML method should provide higher performance results in similar hierarchical datasets. This study is expected to provide higher performance positioning services in indoor spaces in the future. In the future, studies on wireless signal mapping and campus positioning system are planned considering the advantage of the proposed dataset structure and method.

Supplementary Materials: The following are available online at http://www.mdpi.com/2076-3417/9/18/3665/s1. The codes and data are available on https://github.com/acseckin/Hierarchical-Fusion-of-Machine-Learning-Algorithms-in-Indoor-Positioning-and-Localization.

Author Contributions: A.Ç.S. was mainly responsible for the technical aspects and the modeling, as well as the conceived and designed manuscript. A.Ç.S. and A.C. contributed equally in the analysis of the data and writing of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liu, H.; Darabi, H.; Banerjee, P.; Liu, J. Survey of Wireless Indoor Positioning Techniques and Systems. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **2007**, *37*, 1067–1080. [CrossRef]
2. Gu, Y.; Lo, A.; Niemegeers, I. A Survey of Indoor Positioning Systems for Wireless Personal Networks. *IEEE Commun. Surv. Tutor.* **2009**, *11*, 13–32. [CrossRef]
3. Gao, R.; Zhao, M.; Ye, T.; Ye, F.; Wang, Y.; Bian, K.; Wang, T.; Li, X. Jigsaw: Indoor Floor Plan Reconstruction via Mobile Crowdsensing. In Proceedings of the 20th Annual International Conference on Mobile Computing and Networking, Maui, HI, USA, 7–11 September 2014; pp. 249–260.
4. Magrin, C.E.S.; Todt, E. Hierarchical Sensor Fusion Method Based on Fingerprint kNN and Fuzzy Features Weighting for Indoor Localization of a Mobile Robot Platform. In Proceedings of the 2016 XIII Latin American Robotics Symposium and IV Brazilian Robotics Symposium (LARS/SBR), Recife, Brazil, 8–12 October 2016; pp. 305–310.
5. Bahl, P.; Padmanabhan, V.N. RADAR: An In-Building RF-Based User Location and Tracking System. In *Proceedings of the INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*; Tel Aviv, Israel, 26–30 March 2000, Volume 2, pp. 775–784.
6. Yang, Z.; Zhou, Z.; Liu, Y. From RSSI to CSI: Indoor localization via channel response. *ACM Comput. Surv. CSUR* **2013**, *46*, 25. [CrossRef]
7. Goudarzi, S.; Hassan, W.H.; Hashim, A.-H.A.; Soleymani, S.A.; Anisi, M.H.; Zakaria, O.M. A novel RSSI prediction using imperialist competition algorithm (ICA), radial basis function (RBF) and firefly algorithm (FFA) in wireless networks. *PLoS ONE* **2016**, *11*, e0151355. [CrossRef]
8. Wang, X.; Gao, L.; Mao, S.; Pandey, S. CSI-Based Fingerprinting for Indoor Localization: A Deep Learning Approach. *IEEE Trans. Veh. Technol.* **2017**, *66*, 763–776. [CrossRef]
9. Chen, H.; Zhang, Y.; Li, W.; Tao, X.; Zhang, P. ConFi: Convolutional neural networks based indoor wi-fi localization using channel state information. *IEEE Access* **2017**, *5*, 18066–18074. [CrossRef]
10. Jianyong, Z.; Haiyong, L.; Zili, C.; Zhaohui, L. RSSI Based Bluetooth Low Energy Indoor Positioning. In Proceedings of the 2014 International Conference on Indoor Positioning and Indoor Navigation (IPIN), Busan, Korea, 27–30 October 2014; pp. 526–533.
11. Uddin, M.T.; Islam, M.M. Extremely randomized trees for Wi-Fi fingerprint-based indoor positioning. In Proceedings of the 2015 18th International Conference on Computer and Information Technology (ICCIT), Dhaka, Bangladesh, 21–23 December 2015; pp. 105–110.
12. Bozkurt, S.; Elibol, G.; Gunal, S.; Yayan, U. A comparative study on machine learning algorithms for indoor positioning. In Proceedings of the 2015 International Symposium on Innovations in Intelligent SysTems and Applications (INISTA), Madrid, Spain, 2–4 September 2015; pp. 1–8.

13. Xue, W.; Qiu, W.; Hua, X.; Yu, K. Improved Wi-Fi RSSI Measurement for Indoor Localization. *IEEE Sens. J.* **2017**, *17*, 2224–2230. [CrossRef]
14. Akram, B.A.; Akbar, A.H.; Shafiq, O. HybLoc: Hybrid Indoor Wi-Fi Localization Using Soft Clustering-Based Random Decision Forest Ensembles. *IEEE Access* **2018**, *6*, 38251–38272. [CrossRef]
15. Insiteo. Available online: http://www.insiteo.com/ (accessed on 7 July 2018).
16. Wifarer. Available online: http://www.wifarer.com/ (accessed on 7 July 2018).
17. Insoft. Available online: https://www.infsoft.com/ (accessed on 7 July 2018).
18. FIND. Available online: https://www.internalpositioning.com/ (accessed on 7 July 2018).
19. MapsIndoors. Mapspeople. Available online: https://www.mapspeople.com/mapsindoors/ (accessed on 22 August 2019).
20. He, S.; Chan, S.-H.G. Wi-Fi fingerprint-based indoor positioning: Recent advances and comparisons. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 466–490. [CrossRef]
21. Torres-Sospedra, J.; Montoliu, R.; Martínez-Usó, A.; Avariento, J.P.; Arnau, T.J.; Benedito-Bordonau, M.; Huerta, J. UJIIndoorLoc: A New Multi-Building and Multi-Floor Database for WLAN Fingerprint-Based Indoor Localization Problems. In Proceedings of the 2014 International Conference on Indoor Positioning and Indoor Navigation (IPIN), Busan, Korea, 27–30 October 2014; pp. 261–270.
22. Moreira, A.; Nicolau, M.J.; Meneses, F.; Costa, A. Wi-Fi fingerprinting in the real world-RTLS@ UM at the EvAAL competition. In Proceedings of the 2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN), Banff, AB, Canada, 13–16 October 2015; pp. 1–10.
23. Jolliffe, I. *Principal Component Analysis*; Springer: Berlin, Germany, 2011.
24. Ustebay, S.; Gümüş, E.; Aydın, M.A.; Sertbaş, A. Signal Map Reduction for Indoor Localization and Performance Analysis. *J. Fac. Eng. Archit. Gazi Univ.* **2017**, *32*, 1131–1141.
25. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
26. Buitinck, L.; Louppe, G.; Blondel, M.; Pedregosa, F.; Mueller, A.; Grisel, O.; Niculae, V.; Prettenhofer, P.; Gramfort, A.; Grobler, J. API design for machine learning software: Experiences from the scikit-learn project. *arXiv* **2013**, arXiv:13090238.
27. Uddin, J.; Ghazali, R.; Deris, M.M. Does Number of Clusters Effect the Purity and Entropy of Clustering? In Proceedings of the International Conference on Soft Computing and Data Mining, Bandung, Indonesia, 18–20 August 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 355–365.
28. Uddin, J.; Ghazali, R.; Deris, M.M. An empirical analysis of rough set categorical clustering techniques. *PLoS ONE* **2017**, *12*, e0164803. [CrossRef]
29. Uddin, J.; Ghazali, R.; Deris, M.M.; Naseem, R.; Shah, H. A survey on bug prioritization. *Artif. Intell. Rev.* **2017**, *47*, 145–180. [CrossRef]
30. UCI Machine Learning Repository: UJIIndoorLoc Data Set. Available online: https://archive.ics.uci.edu/ml/datasets/ujiindoorloc (accessed on 18 July 2018).
31. Zhang, H. The optimality of naive Bayes. *AA* **2004**, *1*, 3.
32. Murphy, K.P. Naive bayes classifiers. *Univ. Br. Columbia* **2006**, *18*, 60.
33. Altman, N.S. An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *Am. Stat.* **1992**, *46*, 175–185.
34. Masand, B.; Linoff, G.; Waltz, D. Classifying news stories using memory based reasoning. In Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Copenhagen, Denmark, 21–24 June 1992; pp. 59–65.
35. Quinlan, J.R. Induction of decision trees. *Mach. Learn.* **1986**, *1*, 81–106. [CrossRef]
36. Quinlan, J.R. Simplifying decision trees. *Int. J. Man-Mach. Stud.* **1987**, *27*, 221–234. [CrossRef]
37. Avros, R.; Dudka, V.; Křena, B.; Letko, Z.; Pluháčková, H.; Ur, S.; Vojnar, T.; Volkovich, Z. Boosted decision trees for behaviour mining of concurrent programmes. *Concurr. Comput. Pract. Exp.* **2017**, *29*, e4268. [CrossRef]
38. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
39. Liaw, A.; Wiener, M. Classification and regression by randomForest. *R News* **2002**, *2*, 18–22.
40. Akman, M.; Genç, Y.; Ankarali, H. Random forests yöntemi ve sağlık alanında bir uygulama. *Turk. Klin. J. Biostat.* **2011**, *3*, 36–48.

41. Freund, Y.; Schapire, R.E. Experiments with a new boosting algorithm. In Proceedings of the ICML, Bari, Italy, 3–6 July 1996; Volume 96, pp. 148–156.
42. Freund, Y.; Schapire, R.E. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.* **1997**, *55*, 119–139. [CrossRef]
43. Bertoni, A.; Campadelli, P.; Parodi, M. A boosting algorithm for regression. In Proceedings of the International Conference on Artificial Neural Networks, Lausanne, Switzerland, 8–10 October 1997; Springer: Berlin/Heidelberg, Germany, 1997; pp. 343–348.
44. Boser, B.E.; Guyon, I.M.; Vapnik, V.N. A Training Algorithm for Optimal Margin Classifiers. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, Pittsburgh, PA, USA, 27–29 July 1992; ACM: New York, NY, USA, 1992; pp. 144–152.
45. Avros, R.; Volkovich, Z. Detection of Computer-Generated Papers Using One-Class SVM and Cluster Approaches. In Proceedings of the International Conference on Machine Learning and Data Mining in Pattern Recognition, New York, NY, USA, 15–19 July 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 42–55.
46. Müller, K.-R.; Smola, A.J.; Rätsch, G.; Schölkopf, B.; Kohlmorgen, J.; Vapnik, V. Predicting Time Series with Support Vector Machines. In Proceedings of the International Conference on Artificial Neural Networks, Lausanne, Switzerland, 8–10 October 1997; Springer: Berlin/Heidelberg, Germany, 1997; pp. 999–1004.
47. Smola, A.J.; Schölkopf, B. A Tutorial on Support Vector Regression. *Stat. Comput.* **2004**, *14*, 199–222. [CrossRef]
48. Gunn, S.R. Support vector machines for classification and regression. *ISIS Tech. Rep.* **1998**, *14*, 5–16.
49. Özöğür Akyüz, S.; Üstünkar, G.; Weber, G.W. Adapted infinite kernel learning by multi-local algorithm. *Int. J. Pattern Recognit. Artif. Intell.* **2016**, *30*, 1651004. [CrossRef]
50. Kohavi, R. A study of cross-validation and bootstrap for accuracy estimation and model selection. In Proceedings of the IJCAI, Montreal, QC, Canada, 20–25 August 1995; Volume 14, pp. 1137–1145.
51. James, G.; Witten, D.; Hastie, T.; Tibshirani, R. *An Introduction to Statistical Learning*; Springer: Berlin/Heidelberg, Germany, 2013; Volume 112.
52. Nowicki, M.; Wietrzykowski, J. Low-effort place recognition with WiFi fingerprints using deep learning. In Proceedings of the International Conference Automation, Warsaw, Poland, 15–17 March 2017; Springer: Berlin/Heidelberg, Germany, 2017; pp. 575–584.