Pamukkale Univ Muh Bilim Derg, 28(2), 299-312, 2022

Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi

Pamukkale University Journal of Engineering Sciences

A comprehensive review on data preprocessing techniques in data analysis

Veri analizinde veri ön işleme teknikleri üzerine kapsamlı bir inceleme

Volkan ÇETİN[1], Oktay YILDIZ[1]*

[1]Department of Computer Engineering, Faculty of Engineering, Gazi University, Ankara, Turkey.
volkan.cetin2@gazi.edu.tr, oyildiz@gazi.edu.tr

Received/Geliş Tarihi: 21.04.2021 Revision/Düzeltme Tarihi: 11.06.2021 doi: 10.5505/pajes.2021.62687
Accepted/Kabul Tarihi: 07.07.2021 Review Article/Derleme Makalesi

Abstract

With the technological developments, the amount of data stored in the computer environment is increasing very rapidly. Data analysis has become an important research subject for the correct evaluation of these data and to transform them into useful information. Of course, data play an important role in data analysis. However, model performance is highly dependent on the characteristics of the data. For this reason, it is essential to preprocess them before starting any data analysis process. Data preprocessing creates accurate and useful datasets by overcoming erroneous, incomplete, or other unwanted problems. In this study, papers on data preprocessing in the last 5 years have been researched systematically and it has been observed that widely used preprocessing methods are classified under three main branches: data cleaning, data transformation and data reduction. These methods and various algorithms of them are examined, the frequency of use is presented, and comparisons are made in terms of accuracy performance. As the result of the study shows, when data preprocessing methods are not used on raw data or when wrong data preprocessing methods are applied, data analysis methods alone cannot achieve sufficient performance.

Keywords: Data analysis, Data mining, Data preprocessing, Data reduction, Data transformation, Data cleaning, Noise filtering.

Öz

Yaşanan teknolojik gelişmeler ile beraber bilgisayar ortamında saklanan veri miktarı çok hızlı bir şekilde artmaktadır. Bu verilerin doğru bir şekilde değerlendirilmesi ve faydalı bilgiye dönüştürülmesi için de veri analizi önemli bir araştırma konusu olmuştur. Veri analizinde elbette veriler önemli bir rol oynar. Ancak başarım, verinin özelliklerine büyük ölçüde bağımlıdır. Bu sebeple herhangi bir veri analizi süreci başlamadan önce bir ön işlemden geçirmek elzemdir. Veri ön işleme hatalı, eksik ya da istenmeyen diğer sorunların üstesinden gelerek doğru ve kullanışlı veri kümelerini oluşturur. Bu makalede veri ön işleme konusunda son 5 yılda hazırlanmış makale ve bildiriler sistematik olarak araştırılmış ve yaygın olarak kullanılan ön işleme yöntemlerinin üç ana dal altında; veri temizleme, veri dönüştürme ve veri azaltma olarak sınıflandığı görülmüştür. Bu yöntemler ve çeşitli algoritmaları incelenmiş, kullanım sıklıkları sunulmuş ve başarım performansları açısından karşılaştırmaları yapılmıştır. Çalışmanın sonucunun da gösterdiği üzere ham veriler üzerine veri ön işleme yöntemleri kullanılmadığında ya da yanlış veri ön işleme yöntemi kullanıldığında tek başına veri analizi yöntemleri yeterli başarımlara ulaşamamaktadır.

Anahtar kelimeler: Veri analizi, Veri madenciliği, Veri ön işleme, Veri azaltma, Veri dönüştürme, Veri temizleme, Gürültü filtreleme.

1 Introduction

Nowadays, the rapid increase in the amount of data stored in computer environments and the increasing need and difficulty of converting these data into useful information has enabled data analysis solutions to be used frequently. With the emergence of big data as well, data analysis has become more common. Since institutions, organizations and companies are aware of the fact that data analysis has become a vital factor to be competitive, to discover new insights and to personalize their services, they often try to extract information by analyzing the big data they have [1]. Big data refers to the situation where the dataset exhibits various characteristics such as high volume, high variety, and high processing speed of required data [2],[3]. Therefore, with this amount of data, simple statistical methods will either not work, or their performance will remain very low. This need is met by data analysis methods, which have a more complex structure than simple data statistics.

Although there are many different algorithms that can be used as a method in data analysis, basically all of them follow the steps explained below and shown in Figure 1.

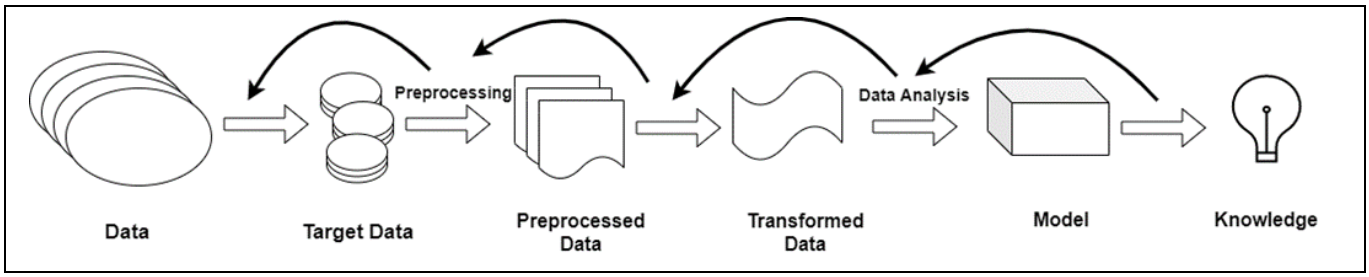

Figure 1. Data analysis steps.

*Corresponding author/Yazışılan Yazar

Data gathering: Without data, there can be no data analysis. Therefore, as a first step, a large amount of data must be gathered regarding the problem to be solved. According to the targets, these data can be gathered from different environments, sources and/or measurements such as banks, surveys, markets, educational environments [4].

Data preprocessing: At this step, which will be examined in detail in the rest of this paper, the collected data are subjected to various processes and made ready to be presented to data mining algorithms. The raw data collected may contain quality issues such as incompleteness, inconsistency, noise, redundancy, and duplicate recording. If these problems are not resolved, the information to be extracted from the data will also be incorrect. Therefore, preprocessing is a very important step in the data analysis process. In the first part of the data preprocessing step, the target data is selected from the entire dataset by performing the data selection process. Then, in the second part, preprocessing methods are applied to fix the errors and improve the quality of the data. In the last part of the data preprocessing phase, this data is transformed into suitable form for data mining algorithms and strengthened for more efficient operation of the mining process and easier understanding of models.

Data mining: Data mining is a process of identifying useful patterns and information from large amounts of data [5]. At this stage, mining methods such as clustering [6] and classification [7],[8] can be applied to the preprocessed data with the help of supervised or unsupervised machine learning algorithms. As a result of these methods, some patterns are searched in the data and a model is created. Using this model, useful knowledge is tried to be extracted. The use of data visualization [9],[10] and information representation [11],[12] tools to represent the data that has been applied to mining methods is also performed at this stage.

The rest of this paper is organized as follows: In the second chapter, preprocessing methods commonly used in data analysis applications are investigated. In this context, researches are limited to journals, books and conference papers for the last 5 years. And then, in the third chapter, the paper is concluded, and the results obtained are evaluated.

2 Preprocessing techniques

Data preprocessing is one of the most important stages of data analysis applications [13]. Raw data often comes with many flaws, such as various inconsistencies, out-of-range values, missing values, noises, and/or excesses. The performance of the learning and mining algorithms to be carried out in the next stages will be weakened due to low quality data [14]. For this reason, the quality of raw data must be increased by passing through various preprocessing stages. Some of the most effective data preprocessing algorithms widely used in data analysis applications are examined under this heading according to their usage, popularity and algorithm behind them.

Basically, there are four main branches of data preprocessing [15]. However, the studies published in the last 5 years on data preprocessing do not contain enough data integration methods for a review. Therefore, the preprocessing methods that have been widely used in the last 5 years can be basically divided into three main branches as shown in Figure 2: data cleaning, data reduction and data transformation. Under the heading of data cleaning; noise filtering and missing value imputation, under the heading of data reduction; feature and instance selections, under the heading of data transformation; normalization and aggregation methods and applications were examined.

2.1 Data cleaning

Data collected in the real world often contain missing and noisy values. Identifying and cleaning these noisy data is one of the challenges of data analytics and skipping this step can lead to inaccurate analysis and unreliable decisions [16]. In order to draw attention to this situation, noise filtering and missing value imputation methods are examined in this section.

2.1.1 Noise filtering

Inaccurate data caused by faulty measurements or human errors within the dataset is called noise. The concept of noise in the data is divided into two: the noise of the measurement value (feature noise) and the noise of the class label [17]. Label noise is known to be more harmful than feature noise, as it is generally more prone to misdirection [18]. Especially the presence of these two types of noise in the training data of classification problems has a great negative effect on decision making and creates serious problems in model production with high accuracy. They affect negatively by extending the model building time, as well. For these reasons, various methods have been developed for noise filtering, which is frequently used in data analysis applications. Due to the reliability of combining more than one method, common noise filtering methods are as follows:

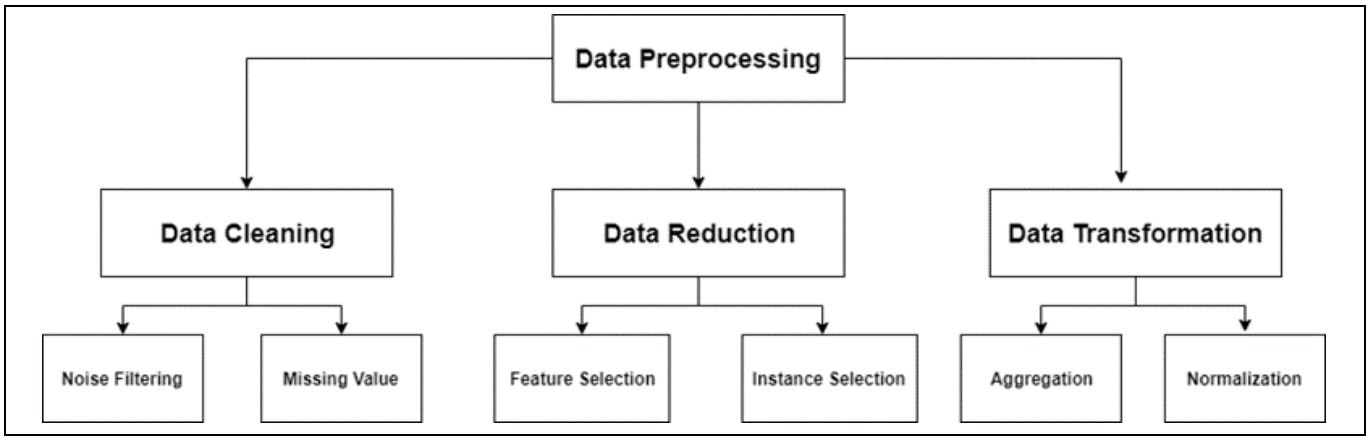

Figure 2. Classification of data preprocessing techniques in data analysis [15].

Ensemble Filter (EF): The use of learning algorithms to eliminate the noise in the dataset can yield successful results. However, instead of using a single learning algorithm for noise detection, when several of them (Artificial Neural Networks [19], Support Vector Machines [20], Decision Trees [21] etc.) are used together, the result is much more successful. This method is called as EF. The most important benefit of the EF technique is that it overcomes the limitations of single methods and can correct the mistakes made by some single methods with the voting mechanism [22].

Iterative Partitioning Filter (IPF): Like EF, IPF also uses multiple classification algorithms for noise filtering by deciding the results by voting method. Difference from EF method; it repeats the data that it filters until a termination threshold is met. In other words, it uses the data that it purified from noise in the previous stages to clean other noises in the later stages. Chen et al. [23] used the IPF method with EasyEnsemble (EE), an under-sampling method to more successfully balance class distribution. IPF and non-IPF EE methods (IPF-EE and EE) were applied on 11 data sets obtained from UCI, in 8 of them IPF-EE produced more successful results than EE method. In other words, it has been shown that EE applied on the noise filtered data with IPF is more successful than the EE applied on the data without filtering. In the same study, SMOTE-IPF was also included for comparison, but it could provide an advantage over IPF-EE in only one dataset. Synthetic Minority Oversampling Technique (SMOTE) is a proposed method for balancing class distribution in data such as EE. But unlike EE, it balances the classes by over-sampling minority-class instances. SMOTE-IPF [24] is presented to further increase the performance of the classification by removing the noise with IPF. This proposed method has been tested on 9 data sets containing unbalanced class distribution, and in all of them SMOTE-IPF has been superior to SMOTE, that is, the situation where noise filtering algorithm is not applied. Results are shown in Table 1.

Table 1. AUC comparison of SMOTE-IPF and SMOTE [24].

Dataset	None	SMOTE	SMOTE-IPF
acl	**88.75**	86.75	88.50
breast	61.73	60.56	**64.40**
bupa	64.40	66.88	**67.53**
cleveland	52.58	54.85	**62.82**
ecoli	72.46	82.16	**86.55**
haberman	57.57	65.41	**66.76**
hepatitis	67.66	71.38	**72.25**
newthyroid	90.87	96.35	**96.63**
pima	70.12	71.29	**73.58**

In a study comparing the IPF and EF noise filtering methods [25], the "banana dataset" in Figure 3, in which it has 5300 samples, 2 classes, 2 attributes, has been selected. Its classes are not separated linearly but are clustered as bananas and it was produced artificially from the KEEL [26] dataset pool. Although the noise filtering results of both algorithms are similar, it has been observed that IPF is more successful than EF in eliminating the samples with high noise in overlapping regions and creating a clearer decision margin.

Machine learning classifiers are not only way to filter the noises. Clustering, which is an unsupervised machine learning method, is also one of the preferred methods. In this method, similar data are grouped in the same cluster as in Figure 4, and the data outside these groups are determined as noise and either deleted or its value is changed to the closest cluster. K-means [27], DBSCAN [28], BIRCH [29] and OPTICS [30] are commonly used clustering algorithms.

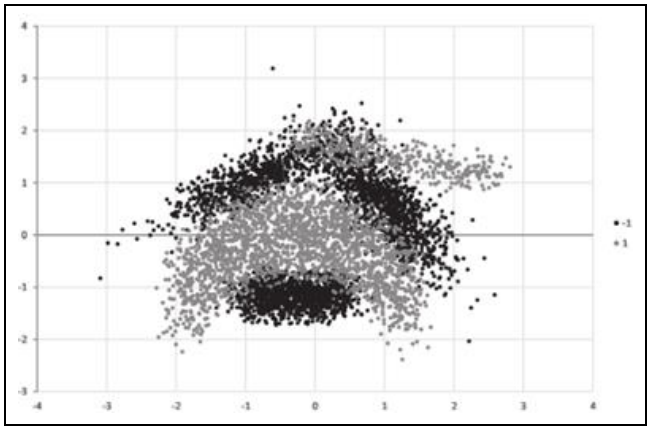

Figure 3. Banana dataset [25].

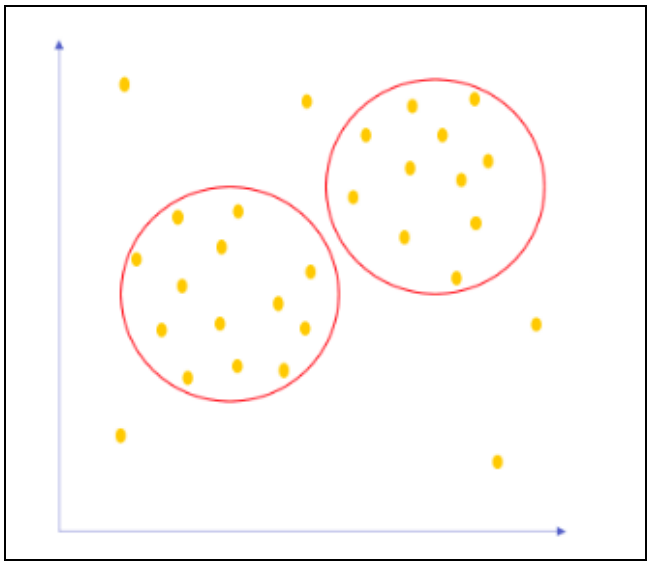

Figure 4. Noise removal using clustering.

Schelling and Plant [31] made improvements to the standard K-means algorithm, which uses clustering method for noise detection, and increased its performance. With this method called K-Means for Noise (KMN), it was observed that when new Voronoi cells were opened at the intersection of Voronoi cells, noisy data points were more easily separated from the clusters. Normalized Mutual Information (NMI) performance, which was 0.82 with K-means, was increased to 0.94 with KMN as a result of the tests performed on a dataset containing 5 clusters and 10% noise. NMI uses entropy to determine the quality of clustering and is a good measure for noise filtering problems using clustering methods. A perfect result with the highest NMI value of 1.0 is unlikely to be achieved, as some of the noise values are contained in a cluster, and 0.94 is almost the best result that can be achieved. In another study [32] presented with a similar purpose, the K-means method was also modified and developed. In this new method, in addition to k clusters, a cluster is created for the noise data. Unlike most existing noise-sensing clustering algorithms, this method assigns all noise values to a group during the clustering process. As a result of experiments on real and synthetic data, better noise filtering performances were obtained again than the standard K-means algorithm.

Another noise elimination method is the binning method. In this method, which has a simpler algorithm compared to other methods, the data is firstly sorted. This sequential data is then divided into equal parts and the data in each piece is changed according to the average or lower/upper values of the part they are in. With this method, changes are also made in the correctly measured data, but noisy values are pulled into the range they should be. Although it is not preferred as much as other methods in the current literature, it is possible to find examples because it is very affected by outlier data and cannot express distorted data well [33].

2.1.2 Missing value imputation

Values for one or more properties of some samples may be missing in the raw dataset. This is called a missing value problem and this situation has various reasons such as the error of the measuring device, the error of the person keeping the record, and the network error. When a dataset having missing values is given to the learning algorithms, either the accuracy rate of the model will decrease, or a model even will not be formed because the algorithm fails. In order to extract knowledge from the dataset, these data must be cleaned and prepared for the data mining process. There are basically three different types of missing data: Missing Completely at Random (MCAR), Missing at Random (MAR), and Missing Data Not at Random (MNAR) [34]. MCAR is when there is absolutely no relationship between missing value and other values observed or missing in the dataset. MAR means that the missingness for a data is not random and is not related to missing data itself but to some of the other observed data in the dataset. The missingness that are not neither MCAR nor MAR is called as MNAR.

One of the ways to overcome the missing value problem in the dataset is to delete the sample with the missing value [35]. However, this situation has some disadvantages. Examples of these disadvantages are the possibility that deleted samples have determinative properties for classification algorithms and that it is not possible to measure that sample again. In addition, the decrease in the number of samples in the dataset can reduce the success of knowledge extraction with data mining. Therefore, considering every sample in the dataset as very valuable, it is the right solution to fill their missing values with appropriate and logical values instead of deleting them. Various missing value imputation methods have been developed to provide this solution:

Mean, Mode, Median: This method calculates the mean, mode or median of the values that are not missing in a column (or feature) and belong to the same class with the missing value and fills in the missing values in the columns separately. Since categorical data cannot be averaged, it can only be used with numerical data. Although it provides very quick and simple methods for missing values, it is a method that underestimates variance and has difficulty in establishing the relationship between variables [36]. It does not take into account correlations between features since it only works at the column level. Due to these disadvantages, it produces more erroneous results compared to other methods and has not been used much in the current literature.

k-Nearest Neighbor (k-NN): In supervised machine learning, the k-NN algorithm works by assuming that the samples belonging to the same class in a dataset are located close to each other. In order to find the class of a new data, the classes of the *k* number of nearest neighbors' data are examined and the most recurring class among these classes is estimated as the class of the new data as shown in Figure 5. This algorithm is also a very useful method for which value to fill a missing value by looking at its *k* number of neighbors. Although the k-NN method is recommended for numerical values in the data set for data filling problems, it is also used for categorical data [37].

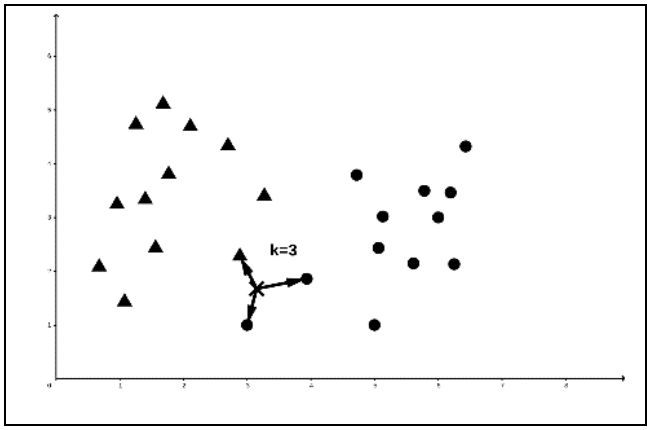

Figure 5. An example of k-NN.

Basically, the proximity relationship is established by calculating the distances between the samples in the dataset and the new data. There are different measures such as Euclid, Minkowski, Hamming, Manhattan and Jaccard to calculate the distance. These measures can give different accuracy rates for different data sets [38]. The most widely used is Euclid, shown in Equation (1). *n* represents the number of dimensions, and *x* and *y* represent two samples. Missing values are determined by considering a certain number of samples, often similar to the situation of interest. The data are classified into groups, and then the missing values are filled in with the corresponding value from the nearest neighbor(s).

$$x = (x_1, x_2, \dots, x_n)$$
$$y = (y_1, y_2, \dots, y_n)$$
$$d(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \quad (1)$$

Choosing the right *k* value is just as important. Usually, different integer values are tested to find the most suitable value, and then the most successful value is selected as *k*. If there are too many noisy values in the dataset, selecting the *k* value small may cause the class of data close to this noisy value and to be estimated incorrectly. Increasing the value of *k* too much can also cause another group, which has different class than new data and larger data than the correct group, to assign the class of new. Although the cross-validation method is frequently used, there are also methods for estimating the *k* value according to the dataset profile [39].

Gene expression data commonly contain an enormous number of missing values. A study using k-NN is presented to fill these missing values [40]. In the method tested on 3 different datasets, the method of filling missing data with mean calculation was also included in the experiments for comparison purposes. As the number of missing data increases, the success rate of the missing value imputation method will decrease with the mean calculation. In datasets containing large amounts of missing data, such as gene expression, it was observed that the filling with mean method achieved

approximately half the success of the k-NN filling method. In another study on the same problem, some improvements were made on k-NN [41]. In addition to the Reduced Relational Grade (RRG) similarity metric, weight coefficients are assigned to neighbors in this method, and missing values can be determined iteratively. As a result of experiments on 5 different datasets, it has been observed that the improved k-NN is more successful than the original k-NN. Lee and Styczynski [42] showed that it is possible to increase the performance by making some improvements in the k-NN method in different problems. In their study, a new k-NN-based algorithm is presented that can further reduce the missing value imputation errors in metabolomic datasets compared to the original k-NN method. As a result, this improved method provides superiority to the original k-NN when the MNAR values reach 20%.

Decision Tree (DT): A decision tree is a structure that divides a dataset containing many samples into smaller subsets by creating a tree structure consisting of nodes, branches and leaves by applying decision rules related to feature values. As a result of this subsetting process, a prediction model is formed that gathers a dataset containing a large number of samples into much smaller groups. It is possible to visualize the DT structure as in Figure 6. Although all data types are categorical in Figure 6, it is also possible to classify numerical data as well by converting them into intervals. In decision tree structure, each feature is represented by a node. The last part of the tree is called the leaf and the classes are shown here. The node at the top is called the root node and the samples are branched (classified), starting from this node, by asking questions according to the rules on the branches, until the nodes or leaves without branches are found [43]. How and in what order this branching process will take place is very important as it will affect the accuracy of the resulting tree.

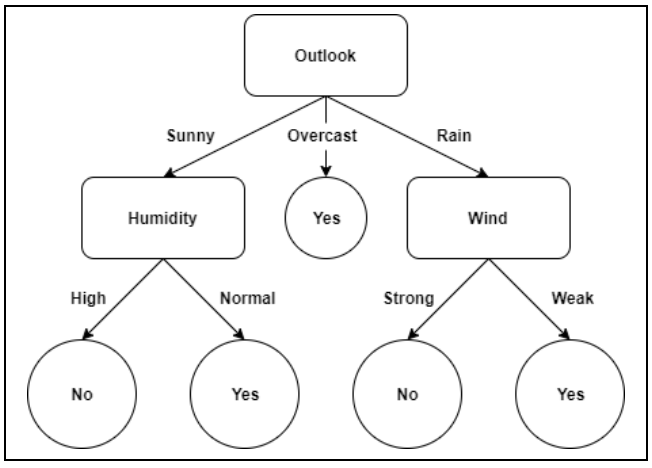

Figure 6. Play tennis example for decision tree [44].

There are different decision tree algorithms used for this purpose. ID3 and the advanced versions of this algorithm, C4.5 and C5.0, are among the most widely used algorithms and work by performing information gain and entropy calculations [45]. The higher the entropy measure, the more uncertain and unstable the result using that attribute, and the highest value equals to 1. Entropy must be low for the information gain to be high. Therefore, the attribute with the least Entropy measure is used in the root node of the decision tree. In the equation where p shows the probability of a group belonging to a certain class, Entropy (H) and information gain are calculated as in Equation (2) and Equation (3), respectively:

$$H(S) = \sum_{i=1}^{n} -p_i \log_2 p_i \tag{2}$$

In the formula in Equation (3), S is the original dataset while D is a divided sub part of it. V is a decision cluster under D.

$$Gain(S, D) = H(S) - \sum_{V \in D} \frac{|V|}{|D|} H(V) \tag{3}$$

In a study where J48, one of the decision tree algorithms, was used to fill the missing values in the cardiovascular dataset, the decision tree method produced more successful results than the k-NN and mean method for missing value imputation [46]. There are 822 samples (different patients) and 22 attributes in the data set. 18 of these 22 features contain missing values between 1% and 30%. It is clearly seen in the study that the dataset obtained by using which missing value method is given to which classification algorithm affects the success of the model. For example, the combination with both the missing value imputation method and the classification algorithm are selected as k-NN, has produced more successful results than the combination where the decision tree is selected as the missing value imputation method and k-NN is selected as the classification algorithm.

Decision tree and k-NN are two classification methods that are often compared within the scope of missing value. In the performance analysis conducted by Abidin et al. [47], the machine learning classification method that stands out is the decision tree. Although the Bayesian network and the decision tree have almost the same results, the decision tree gives better results for large data sets with a greater number of missing values. As a result of the analyzes performed on 10 different datasets, k-NN came to the fore as the most unsuccessful missing value imputation method. Classification accuracy rate, Mean Squared Error (MSE) [48], Root Mean Square Error (RMSE) [48],[49] and Mean Absolute Error (MAE) [49] are generally preferred as metrics for success criteria. In another performance analysis performed on 5 different datasets [50], it has been observed that the C5.0 decision tree method fills the missing values more successfully than the other two methods, the k-NN algorithm also gives good results, but the calculations take a lot of time in large datasets, and mean method can give good results only if the percentage of missing value is below 5%. Error rates for the 4 datasets used in the study are shown in Table 2. In the experiment, Iris dataset has 4 features, 150 instances and has 10% missing value ratio. Adult dataset has 13 features, 30162 instances and has 20% missing value ratio. Wine dataset has 13 features, 4898 instances and has 25% missing value ratio. And Credit dataset has 16 features, 690 instances and has 15% missing value ratio. Since the C5.0 algorithm has the possibility to detect and extract the redundant features, it uses less features in the classification stage compared to the other two methods.

Table 2. Classification error rates [50].

Dataset	Imputation with mean or mode	Imputation with k-NN	Imputation with C5.0
Iris	0.16	0.09	**0.04**
Adult	0.17	0.17	**0.15**
Wine	0.47	0.49	**0.36**
Credit	0.24	0.17	**0.14**

2.2 Data reduction

Today, the data produced by various sensors and applications are growing rapidly on both row (instance) and column (feature) basis. This creates a bottleneck for data analytics and increases the load on machine learning and data mining algorithms [51]. Not only does it increase complexity and prolong the time to obtain results, but it may also even prevent data analysis algorithms from extracting accurate information due to unnecessary and irrelevant data it contains. Therefore, it is necessary to reduce the size of these data and not to reduce their quality while doing this. In order to overcome such problems, feature selection and instance selection methods, which are among the data reduction methods, will be examined under this section by providing their usage areas and related comparisons in the current literature.

2.2.1 Feature selection

Feature selection is a preprocessing technique that defines key attributes of a problem. Basically, it is achieved by reducing the number of features, namely the number of columns in a dataset. When the number of features is reduced without reducing the quality of the dataset, the model performance rate and inference quality increase, while the learning time and space required for storage are reduced. Various feature selection algorithms are available to provide these benefits. These algorithms are basically divided into three categories: filters, wrappers, and embedded methods.

Filters: Filter methods calculate the contribution of the columns in the dataset to reaching the result with a scoring mechanism, using statistical calculations. If the value created as a result of these calculations is below a specified threshold value, it is not included in the next data analysis steps, if it is above, the column belonging to that value is selected as a feature. It is independent of any learning method as it focuses on the general characteristics of the data. Their calculation costs are considered low, and their generalization capacities are good.

Relief algorithm is one of the most widely used filter methods for feature selection. Relief calculates a numerical index that evaluates the significance of the feature or the level of association with respect to the observed output directly from the data [52]. It uses the method of giving weight value (W) to each column vector (A) while calculating this index. Relief algorithm can be seen in Algorithm 1. The two closest neighbors labeled with different classes are predicted for each given instance, one called near-hit and the other called near-miss.

Algorithm 1. Relief.

Algorithm inputs: Column (feature) vectors, class values, threshold.

Algorithm output: Column vectors whose weights are beyond the threshold (W).

1. Assign 0 to all weight values. W[A]=0
2. **for** i=1 to m **do**
3. Choose a random instance (R_i).
4. Find near-hit (H) and near-miss (M).
5. **for** A=1 to a **do**
6. W[A]=W[A]-diff(A, R_i,H)/m + diff(A, R_i,M)/m
7. **end**
8. **end**
9. Return features whose weight (W) is bigger than threshold.

Tripathi and Trivedi [53] performed sentiment analysis of Indian cinema through various feature selection methods and compared to other methods, the best result was obtained with a Relief-based method with an F-value of 88.8%. In another study [54], a Relief-based feature selection algorithm was proposed to select the most efficient feature combination in DNA microarray data with high dimension but small number of samples. Experiments have shown that the proposed algorithm eliminates relatively less relevant features in the dataset and positively affects the classification performance in terms of both accuracy and time.

Another widely used filter method for feature selection is the Correlation-based Feature Selection (CFS) algorithm. CFS is a simple filter algorithm that sorts subsets of features based on a correlation-based heuristic evaluation function. The CFS method assumes that features having little relevance with the class show a low correlation and should therefore be ignored by the algorithm. The criterion used to find the most suitable subset in a dataset containing l number of features can be expressed as follows [55]:

$$M_s = \frac{l\overline{t_{cf}}}{\sqrt{l + l(l-1)\overline{t_{ff}}}} \quad (4)$$

In Equation (4) M_s is the subset (S) with l features, $\overline{t_{cf}}$ is the average correlation value between features and class labels, $\overline{t_{ff}}$ is the average correlation value between two features. There are many different usage examples of CFS in the current literature. In one of these [56], CFS was used to provide image recognition of apple diseases based on color, shape and texture features obtained from diseased apple leaf images. In this study, it has been shown that when CFS method is used together with Genetic Algorithm (GA), more effective feature subsets can be selected. A very comprehensive study was conducted by Amarnath and Balamurugan [57] to show how effective CFS is. 6 different feature selection methods were applied on 15 different data sets from UCI, the number of features in the result subset is shown in Table 3 and the accuracy rates obtained when this subset is trained with Naive Bayes (NB) are shown in Table 4. As the results show, in general, the CFS method provided more successful results compared to other feature selection methods in terms of both the success of selecting the subset with the least feature and its effect on the model success.

Wrappers: In wrapper methods, feature selection is evaluated using a learning algorithm. The learning outcome is applied iteratively according to the success of the model and the optimum subset is selected. Since it uses machine learning algorithms in sub-feature set selection, it generally produces more successful results than filters, but it is slower. The two most common methods in the current literature are Sequential Forward Selection (SFS) and Sequential Backward Selection (SBS).

SFS is a wrapper method that starts with an empty feature set and gradually adds selected features with the help of some evaluation methods. SFS, whose pseudo code is included in Algorithm 2, is a bottom-up method as it reaches the final set from an empty set (Y_0) by adding new attributes (x_{new}) step by step. The most important criterion for the attribute to be added at each level is to increase the current accuracy rate ($\arg\max[J(Y_k + x)]$). If adding any of the remaining features does not affect the success of the model positively, the iteration will stop, and the feature subset will be finalized.

Table 3. Selected feature count [57].

Dataset	Instance count	Feature count	CFS	Chi square	GR	IG	One R	SU
C lens	24	5	1	2	2	2	3	1
S landing	15	7	2	6	3	5	4	4
DNA P	106	58	6	6	6	6	5	5
TTT	958	10	5	1	1	1	1	1
Parity	100	11	3	6	6	6	6	6
Nursery	12960	9	1	1	1	1	5	5
Adult	20	5	2	2	2	2	2	2
Chess	2128	37	6	7	5	7	5	9
Monk	124	6	2	2	2	2	2	2
Weather	14	5	2	2	2	2	3	2
Splice	3190	61	22	7	8	8	9	2
S heart	267	23	12	8	10	9	19	10
K.R. vs K.P.	3196	37	7	11	15	11	19	10
Car-Eval	1728	7	1	6	6	6	6	6
Balloon	20	5	2	2	2	2	2	2

Table 4. NB accuracy (%) with selected features [57].

Dataset	CFS	Chi square	GR	IG	One R	SU
C lens	70.83	**87.50**	**87.50**	**87.50**	54.17	70.83
S landing	**80.00**	73.33	**80.00**	73.33	73.33	73.33
DNA P	95.28	95.28	95.28	95.28	95.28	**95.34**
TTT	**72.44**	69.94	69.94	69.94	69.94	69.94
Parity	**50.00**	46.00	46.00	46.00	47.00	46.00
Nursery	70.97	70.97	70.97	70.97	**88.84**	70.97
Adult	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Chess	**94.45**	89.61	92.34	89.61	86.33	90.23
Monk	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Weather	**78.57**	**78.57**	**78.57**	**78.57**	71.43	**78.57**
Splice	**96.14**	93.89	94.17	94.17	94.29	94.17
S heart	**82.02**	76.78	80.15	79.03	79.03	79.03
K.R. vs K.P.	**91.99**	88.17	89.86	89.11	88.11	88.67
Car-Eval	70.02	**85.53**	**85.53**	**85.53**	**85.53**	**85.53**
Balloon	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**

Although it produces faster results than other wrapper methods, it does not always produce the best accuracy result, as there is no option to subtract one of the features it adds at any stage and replace it with a possible better feature.

Algorithm 2. SFS [58].

1. Start with an empty set.
   $Y_o = \{\emptyset\}$
2. Choose the feature which gives the best accuracy.
   $x_{new} = \arg\max[J(Y_k + x)]\,, x \notin Y_k$
3. Update feature subset.
   $Y_{k+1} = Y_k + x_{new}; k = k + 1$
4. Go to the second step.

Pasyuk et al. [59] compared different sequential feature selection methods to use in network traffic flow classification problem. These methods are: SFS, SBS, Sequential Forward Floating Selection (SFFS), and Sequential Backward Floating Selection (SBFS). k-NN, Random Forest and Gradient Boosting classification algorithms have been chosen to be used with these methods. As a result of the experiments performed on the dataset containing 30 features and 28673 samples, it was observed that the SFS and SFFS methods were more successful in terms of accuracy compared to the back selection methods for the network traffic flow classification problem. Comparing to the SFS method, the SFFS method provides opportunity to remove the feature that was added in previous step. The best result was the combination of k-NN and SFFS. In another study using SFS method as feature selection, a solution for facial expression recognition problem is presented [60]. The Extended Cohn-Kanade dataset was used as the dataset, and 2278 features were extracted using the distances of the landmark points in the face. Then, SVM was used as the classification algorithm and a classification success of 89.9% was achieved in addition to finding the effective features in determining facial expressions and emotion.

SBS, on the other hand, starts with a set (X) containing all the features instead of starting with an empty feature set with the opposite working principle of the SFS method and continues recursively as shown in Algorithm 3 by removing the features ($Y_k - x_{worst}$) according to their effect ($J(Y_k - x)$) on the model success.

Widiyanti and Endah [61] used SBS, SFS and Relief in the preprocessing stage of the musical emotion recognition problem and compared the results.

Algorithm 3. SBS.

1. Start with the original dataset.
   $Y_o = X$
2. Find the feature that causes the worst accuracy.
   $x_{worst} = \arg\max[J(Y_k - x)], x \in Y_k$
3. Update the feature subset by removing the feature found at the previous step.
   $Y_{k+1} = Y_k - x_{worst}; k = k + 1$
4. Go to the second step.

In the experiments, when the SBS and SFS methods were used with the SVM classifier, they gave equal accuracy with each other and higher than the Relief method. However, SBS has been the recommended method in the study because it contributes to the production of faster models by selecting a smaller number of features compared to SFS. Yulianti and Saifudin [62] compared SFS, SFFS, SBS and SBFS methods using the Naive Bayes classification method for customer churn estimation. As a result of the experiments conducted on the Telco Customer Churn dataset with 20 features and 7043 samples, it was observed that the best performance rates were provided by the backward methods, namely SBS and SBFS. Both methods decreased the number of features from 29 to 19, in addition to increasing the success of the model, as well as reducing the calculation time. Wang et al. [63] combined SBS with Multi-Layer Perceptron (MLP) to select the most efficient DDoS features in the NSL KDD dataset containing 41 features. As a result of the experiments conducted with SBS MLP, SFS MLP and MLP without feature selection, it was observed that the SBS MLP method obtained the subset with the least number of features and provided a better accuracy performance than both the original feature set and the feature set obtained with SFS MLP. While SFS decreased the number of 41 features from 41 to 35 and achieved a success of 97.61%, SBS achieved a success performance of 97.66% by reducing the number of features to 31.

Embedded Methods: Embedded methods incorporate machine learning methods into their algorithms simultaneously and directly to perform feature selection operations. In other words, unlike wrapping methods, it does not make the feature selection by first finding a subset of features and then creating and evaluating a model with machine learning methods; it uses machine learning methods directly to calculate the most efficient feature subset. This is why it is called the embedded method. Wrapper methods evaluate each subset of features it creates with classification methods, but because the embedded methods do this all at once, they are much closer to filter methods in speed. Since they make use of machine learning, they also produce more accurate results than filter methods. The most common methods in the current literature are Lasso and Ridge.

Lasso and Ridge are called regularization-based embedded methods. Lasso uses the L1 regularization, and the Ridge uses the L2 regularization [64]. L1 regularization tries to minimize the absolute value of the feature coefficient sums to zero, while L2 regularization tries to minimize the square of the coefficient sums. In a study presented for the bug prediction problem in software [65], Elastic Net methods using L1 and L2 arrangements together, Lasso and Ridge were compared as feature selection methods. As a result, it has been observed that all three methods have almost the same but very important effect on model success. In a study aimed at predicting heart disease [66], Lasso and Ridge methods were compared as feature selection. The "Cleveland heart" dataset with 72 features was used as the dataset obtained from UCI as open access. As a result of the experiments performed using different classification methods, it is shown in Table 5 that Lasso method provides more successful accuracy percentages compared to the Ridge method.

Table 5. Heart disease prediction performance (%) [66].

Classification method	No feature selection	Lasso	Ridge
Random forest	47.02	84.98	**85.31**
Extra trees	55.83	**90.32**	84.77
Gaussian NB	57.17	**94.92**	**94.92**
Logistic regression	40.73	**63.73**	59.12

2.2.2 Instance selection

It is not just the features that take up irrelevant and redundant space in the raw datasets. Among the instances, each expressed in a row, there may also be those that negatively affect model success and/or time performance. In such cases, using the appropriate instance selection method is at least as important as feature selection. Instance selection is the process of selecting a subset of an original dataset by finding the instances that best represent the dataset, without reducing model success. In this way, it is ensured that data mining methods can be applied on large datasets. Algorithms adopting the Nearest Neighbor (NN) method are widely used in the literature. Condensed Nearest Neighbor (CNN) [67] and Edited Nearest Neighbor (ENN) [68] are two of these algorithms that stand out.

The CNN method is an instance selection method that examines the close neighbors of the instances and reduces the number of them without compromising the original model accuracy. In the CNN, whose algorithm is seen in Algorithm 4, the first step is to start with an empty instance set (Z). Another instance (x'), which is located very close to a randomly selected instance (x) from the original dataset (X), but whose class is different, is searched. If found, this instance means that it is quite possible that it is on a boundary separating classes. That instance is then added to the sub dataset (Z) that will be used for later classification. If the instance and the instance very close to it have the same class, this means that the selected instance does not provide useful information other than the information we already have in the Z set, and it is not added to the subset.

Algorithm 4. CNN.

1. Start with an empty instance subset (Z).
   $Z = \{\}$
2. Choose a random instance (x ∈ X)
   Find $x' \in Z$ satisfying the equation $||x - x'|| = min_{x^j \in Z} ||x - x^j||$. If class($x$) is not equal to class($x'$), add instance x to Z.
3. Repeat the second step until Z does not change anymore.

The ENN method is a method that selects the instances according to the class values of the closest neighbors like CNN. But unlike CNN, instead of starting with an empty sample set, it starts with the original dataset and examines the k number of neighbors of each instance. If the class of the majority of these k instances and the class of the selected instance is different, it

creates a subset by removing that instance from the dataset. In this way, both noisy instances are removed from the dataset and the boundaries separating the instances belonging to different classes become clearer. Algorithm 5 illustrates the ENN algorithm.

A comparison of ENN and CNN is included in the study presented by Kasemtaweechok and Suwannik [69]. In the experiments performed on 9 data sets taken from UCI and KEEL and whose properties are shown in Table 6, the *k* value for both methods is selected as 3 and the distance metric is Euclid. In the experimental results, CNN provided an average of 74.3% accuracy and 0.43 kappa value, while ENN was able to produce more successful results than CNN with 79.41% accuracy and 0.46 kappa value. Confusion matrix is used to obtain the kappa value. Kappa value varies between -1 and +1, while -1 represents the biggest difference, i.e. failure, +1 represents the greatest measure of success [70].

Algorithm 5. ENN.

1. Start with the original dataset (Z).

   $Z = X$

2. If the class of a randomly selected instance (x) and the majority of the k number of neighboring instances are different, remove it from Z set.

   Choose a random instance ($x \in Z$)

   if class(x) != class(kNN(x))

   remove x from Z

3. Repeat the second step until Z does not change anymore.

Table 6. Datasets used for comparison of ENN and CNN [69].

Dataset	Number of instances	Number of features	Number of classes
Nursery	12690	8	5
Magic	19020	10	2
Letter	20000	16	26
Bank	45211	16	2
Adult	45222	14	2
Shuttle	58000	9	7
Fars	100968	29	8
Census	142521	41	3
Skin	245057	3	2

A similar study was carried out by Song et al. [71]. In the study, threshold-based CNN (TE-CNN), threshold-based ENN (TE-ENN), discretization-based CNN (DE-CNN) and discretization-based ENN (DE-ENN) methods were tested on 19 data sets from KEEL. Coefficient of determination (R^2) was used as the measurement of success and the results are shown in Table 7.

2.3 Data transformation

Even if the problems caused by noise, missing value and unnecessary features have been eliminated by using data cleaning and data reduction methods on raw data, this processed new dataset may not be suitable for analysis by a data analysis application. This inappropriately structured data may cause the performance and efficiency of the data mining model to decrease. The methods that transform data into the appropriate format for data mining algorithms are called data transformation methods. Normalization and data aggregation are two important data transformation methods.

Table 7. Datasets used in comparison of TE-ENN, DE-ENN, TE-CNN and DE-CNN and R^2 results [71].

Dataset	Instance count	Feature count	TE-CNN	TE-ENN	DE-CNN	DE-ENN
Abalone	4052	8	**0.739**	0.712	0.635	**0.739**
Airfoil	1503	6	0.482	**0.496**	0.462	0.469
ANACALT	4052	7	0.992	0.992	0.940	**0.993**
California	20640	8	0.561	0.555	0.474	**0.563**
CASP	45730	9	**0.921**	0.920	0.919	**0.921**
CCPP	9568	4	**0.975**	0.963	0.973	**0.975**
Compaic	8192	21	**0.941**	0.940	0.937	0.935
Concrete	1030	8	0.855	0.762	**0.862**	0.732
Ele2	1056	4	0.996	0.996	**0.997**	0.996
Friedman	1200	5	**0.953**	0.943	0.944	0.952
House	22784	16	0.314	**0.364**	0.315	0.301
Mortgage	1049	15	**0.996**	0.959	**0.996**	0.983
Plastic	1650	2	0.874	0.845	0.861	**0.879**
Pole	14998	26	**0.921**	0.856	0.880	0.912
Quake	2178	3	0.122	0.138	**0.178**	0.169
Tic	9822	85	**0.195**	0.190	0.115	0.115
Treasury	1049	15	**0.994**	0.952	0.992	0.984
Wankara	1609	9	0.990	0.988	**0.991**	0.989
Wizmir	1461	9	0.994	**0.996**	**0.996**	**0.996**

2.3.1 Normalization

Normalization is the scaling of the data of a feature to certain intervals such as [-1.0, 1.0] or [0.0, 1.0] and is usually required when there are features at very different scales in a dataset. Otherwise, there may be a decrease in the effectiveness of another, equally important, but lower-scaled feature due to other features with values on a much larger scale. This will negatively affect the accuracy performance of the data mining model. For this reason, the normalization process is applied to the features to bring them to the same scale. Min-max normalization, z-score normalization and decimal scale normalization are the three most common methods.

Min-max normalization: Calculation is made according to the difference between the smallest and largest values of the data to be normalized. In Equation (5), *min* shows the smallest value in the values of the feature, *max* the largest, v the value to be normalized, and new_{max} and new_{min} show the new range to be normalized.

$$v' = \frac{v-min}{max-min}(new_{max} - new_{min}) + new_{min} \quad (5)$$

Z-score normalization: Values are normalized based on mean and standard deviation calculations. In Equation (6), $\bar{A}$ represents the mean value and σ_A represents the standard deviation.

$$v' = \frac{v - \bar{A}}{\sigma_A} \quad (6)$$

Decimal scale normalization: Applies the normalization process by moving the decimal point of the values. As seen in Equation (7), this movement of decimal points depends entirely on the maximum value among all values in the feature. The value j is the smallest number that satisfies the inequality of max(|v'|) <1.

$$v' = \frac{v}{10^j} \quad (7)$$

In a study comparing these three normalization methods, the effects of various normalization methods on the prediction of stock market movements were investigated [72]. In the study, both [0,1] (Min-Max-1) and [-1,1] (Min-Max-2) ranges were used in comparisons for Min-Max normalization. In addition, the Median and Median Absolute Deviation (MMAD) normalization method is also included. MMAD normalization is similar to Z-Score normalization but uses the median and median absolute deviation to normalize data values rather than mean and standard deviation [72]. It is shown in Table 8 that different normalization methods with SVM classification applied on 9 datasets and performed the best result for different datasets.

Table 8. Comparison of normalization accuracy (%) [72].

Dataset	No normalization	Min-Max-1	Min-Max-2	Z-score	Dec. scale	MMAD
1	**60.70**	56.22	56.72	59.20	55.72	57.71
2	60.29	56.37	57.35	**60.78**	57.35	59.31
3	60.30	57.79	60.30	**61.31**	57.79	59.80
4	59.22	**60.68**	**60.68**	58.25	58.25	58.74
5	60.40	59.41	59.90	61.39	60.40	**62.87**
6	63.46	63.46	63.46	63.46	**65.38**	64.42
7	**64.18**	**64.18**	**64.18**	63.18	63.18	63.68
8	66.67	65.69	66.18	65.69	65.20	**67.65**
9	66.17	66.17	66.17	66.17	64.18	**67.66**

In another study comparing normalization methods, it was observed that mean and standard deviation measurements are more suitable for data normalization compared to min-max and median measurements [73]. Z-score, which is a mean and standard deviation normalization method, provided good classification performances and was able to overcome outliers more effectively than other normalization methods. Pandey and Jain [74] examined the effect of different normalization methods on k-NN performance in the Iris dataset. Classification results compared with different k values are shown in Table 9.

Table 9. Comparison of normalization accuracy (%) with different k values of k-NN [74].

k value in k-NN	Min-max	Z-score
1	**100**	85.71
13	**95.23**	85.71
50	90.47	**100**
100	**66.67**	42.85

Eesa and Arabo [75] performed a similar study for back propagation neural networks. As a result of the experiments performed on 8 different data sets, the Median and Median Absolute Deviation normalization method provided the best result in 4 out of 8 datasets and the third best result in the remaining 4 datasets. Ali and Senan [76] investigated the effect of normalization on video classification performance. As a result of the tests performed with the MLP classification method on the VSD2014 [77] dataset obtained from Technicolor Group, the best classification success was achieved with a large difference with the min-max normalization method and the results are reflected in Table 10.

Table 10. Comparison of normalization accuracy (%) over VSD2014 [76].

MLP Hidden Node Count	Min-max [0,1]	Min-max [-1,1]	Z-score
5	**97**	57	49
10	**98**	55	53
20	**97**	59	50

2.3.2 Aggregation

Data aggregation is the process of presenting data in a summarized form. It is realized by gathering two or more attributes under a single attribute. It plays an important role in converting the data collected from different sources into the appropriate format. It not only transforms data but also reduces the size of the data set, making the use of memory and time more efficient.

Aggregation method is widely used in systems where a large amount of data is collected from many different sensors, such as the Wireless Sensor Network (WSN). In a study [78], aggregation was made using the similarity function, one-way Anova model and distance functions. Morell et al. [79] also, presented a solution to the same problem with the Principal Component Analysis (PCA). PCA is a transformation technique that transforms the dataset into less associated variables, allowing it to be reduced to a smaller size. Xie et al. [80] proposed a multi-scale PCA to detect faults on data gathered from WSN. This method allows gathering data from WSNs with various time and frequencies thanks to its multi-scale feature. Li et al. [81] used PCA for data aggregation in WSN to minimize the total amount of wireless sensor data. When PCA is implemented recursively by updating the parameters in each iteration, accuracy of data aggregation performance can be increased for data analysis [82]. Thanks to these advantages, PCA is widely used in WSN problems as aggregation of data [83]. Moreover, considering the time elapsed and memory and CPU usage, PCA is shown that it is more efficient algorithm comparing to other dimension reduction methods such as Isomap, L-Isomap, Laplacian Eigenmaps, FastMVU, SNE and t-SNE [84]. In addition to PCA, Replication Filtering methods are also shown among the alternatives that can be used to securely aggregate wireless sensor networks [85].

3 Conclusions

In raw data, incomplete, inconsistent, unnecessary, noisy and outlier data are often included due to the measuring device or human errors. For this reason, the preprocessing stage is very important in data analysis. When these preprocessing steps are not followed carefully, the efficiency of data analysis applications will decrease, and the extraction of knowledge will be difficult. In this study, data cleaning, data transformation and data reduction methods, which are the three main branches of preprocessing, were examined in the literature by following up-to-date papers which are published in the last 5 years. Although there are plenty of methods presented in the literature for each data preprocessing branch for data analysis, widely used state-of-the-art methods which lead high model accuracies are included in this study. Moreover, comparisons of these methods in terms of accuracy performance are provided. The studies examined are listed in Table 11.

Table 11. Studies selected for review and preprocessing field.

Data cleaning	Data reduction	Data transformation
X. Chen et al. [23], JA. Sáez et al. [24], S. García et al. [25], B. Schelling, C. Plant [31], G. Gan, M. Kwok-Po Ng [32], B. Cigdem et al. [33], H. de Silva and A. S. Perera [40], Y. He and Pi. Dechang [41], JY. Lee, MP Styczynski [42] D. Davis and M. Rahman [46], N.Z. Abidin [47], T. Aljuaid and S. Sasi [50],	A. Tripathi and S. K. Trivedi [53], M. Liu et al. [54], Z. Chuanlei et al. [56], dB. Amarnath, S. Balamurugan [57], A. Pasyuk et al. [59], C. Gacav et al. [60], E. Widiyanti and S. N. Endah [61], Y. Yulianti and A. Saifudin [62], M. Wang et al. [63], H. Osman et al. [65], D. Panda [66], C. Kasemtaweechok and W. Suwannik [69], Y. Song et al. [71]	J. Pan et al. [72], D. Singh and B. Singh [73], A. Pandey and A. Jain [74], A. Eesa and W. Arabo [75], A. Ali and N. Senan [76], H. Harb et al. [78], A. Morell et al. [79], Y. Xie et al. [80], J. Li et al. [81], T. Yu et al. [82], S. Boubiche et al. [83], K. Yildiz et al. [84], E. Choudhari et al. [85]

Within the scope of data cleaning, noise filtering and missing value imputation methods are reviewed. For noise filtering, since EF and IPF techniques make use of several machine learning algorithms, noise filtering performance of them are superior to the ones that do not use machine learning methods. Mean calculation, k-NN and decision tree methods were examined for missing value imputation. Although mean calculation method runs fast and are easy to implement, the model performance will decrease as the missing value ratio increases since it does not regard the relationship with other features in the dataset. k-NN is also easy to implement comparing to other classification algorithms, but it is affected badly by noise and outlier data. Decision trees, on the other hand, provide the best accuracies as well as they are suitable for all data types.

Data reduction phase can be basically divided into two methods as feature selection and instance selection. The feature selection method is performed by reducing the number of features, namely columns, in a dataset. When the number of features is reduced without reducing the quality of the dataset, the model performance and quality increase, while the learning time and space required for storage are reduced. Therefore, it is considered as a very important preprocessing step. There are three main methods of feature selection: filters, wrappers, and embedded methods. Wrappers and embedded methods provide more efficient results in data analysis problems as they accommodate machine learning methods. Comparing to wrapper methods, embedded methods are less prone to overfitting and have better model performance. In the instance selection, which is another data reduction category, data reduction is performed over instances, not attributes. For this method, the closest neighbors are searched, and unnecessary samples are removed from the dataset. The two most widely used examples in the current literature are CNN and ENN.

The third and final data preprocessing method is data transformation. With this method, the values in the dataset are converted into suitable and better formats for data analysis algorithms. Normalization and aggregation are the two preferred methods in this context. Min-max, Z-score and decimal scale normalization are the three most widely used algorithms in the current literature. To minimize and aggregate the total amount of data, PCA is usually preferred especially in WSN problems.

As a future study, by making improvements on the algorithms reviewed in this study, new solution approaches can be proposed. Because this study also reveals that better model performances for data analysis can be obtained by improving the standard data preprocessing methods.

4 Author contribution statements

In the scope of this study, Volkan ÇETİN contributed to the formation of the idea, writing, literature review, assessment of the obtained results from the literature, Oktay YILDIZ contributed to formation of title and abstract, editing the article in terms of content, assessment of the obtained results from the literature.

5 Ethics committee approval and conflict of interest statement

There is no need to obtain permission from the ethics committee for the article prepared.

There is no conflict of interest with any person / institution in the article prepared.

6 References

[1] Oussous A, Benjelloun F, Lahcen A, Belfkih S. "Big data technologies: a survey". *Journal of King Saud University-Computer and Information Sciences*, 30(4), 431-448, 2018.

[2] Choi TM, Wallace SW, Wang Y. "Big data analytics in operations management". *Production and Operations Management*, 27(10), 1868-1883, 2018.

[3] García S, Ramírez-Gallego S, Luengo J, Benítez JM. "Big data preprocessing: methods and prospects". *Big Data Analytics*, 1(1), 1-22, 2016.

[4] Anoopkumar M, Rahman AMJMZ. "A Review on data mining techniques and factors used in educational data mining to predict student amelioration". *2016 International Conference on Data Mining and Advanced Computing*, Ernakulam, India, 16-18 March, 2016.

[5] Yıldırım P, Birant D. "Application of data mining techniques in cloud computing: a literature review". *Pamukkale University Journal of Engineering Sciences,* 24(2), 336-343, 2018.

[6] Venkatkumar IA, Shardaben SJK. "Comparative study of data mining clustering algorithms". *2016 International Conference on Data Science and Engineering*, Cochin, India, 23-25 August 2016.

[7] Çığşar B, Ünal D. "Comparison of data mining classification algorithms determining the default risk". *Scientific Programming*, 2019, 1-8, 2019.

[8] Umadevi S, Marseline KSJ. "A survey on data mining classification algorithms". *2017 International Conference on Signal Processing and Communication*, Coimbatore, India, 28-29 July 2017.

[9] Ajibade S, Adediran A. "An overview of big data visualization techniques in data mining". *International Journal of Computer Science and Information Technology Research*, 4(3), 105-113, 2016.

[10] Kunjir A, Sawant H, Shaikh NF. "Data mining and visualization for prediction of multiple diseases in healthcare". *2017 International Conference on Big Data Analytics and Computational Intelligence*, Chirala, India, 23-25 March 2017.

[11] Zhou X, Yang C, Meng N. "Method of knowledge representation on spatial classification". *2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, Tianjin, China, 14-16 August 2009.
[12] Guowei Y, Xinghua L, Xuyan T. "A new knowledge representation based matter element system and the related extension reasoning". *International Conference on Natural Language Processing and Knowledge Engineering, 2003,* Beijing, China, 26-29 October 2003.
[13] García S, Luengo J, Herrera F. *Data Preprocessing in Data Mining*. 1st ed. New York, USA, Springer, 2015.
[14] Ramírez-Gallego S, Krawczyk B, García S, Wozniak M, Herrera F. "A survey on data preprocessing for data stream mining: Current status and future directions". *Neurocomputing*, 239, 39-57, 2017
[15] Malik JS, Goyal P, Sharma AK. "A comprehensive approach towards data preprocessing techniques & association rules". *Proceedings of the 4th National Conference*, Delhi, India, 25-26 February 2010.
[16] Chu X, Ilyas I, Krishnan S, Wang J. "Data cleaning: Overview and emerging challenges". *SIGMOD 16: Proceedings of the 2016 International Conference on Management of Data*, San Francisco, USA, 26 June-01 July 2016.
[17] Pelletier C, Valero S, Inglada J, Champion N, Marais Sicre C, Dedieu G. "Effect of training class label noise on classification performances for land cover mapping with satellite image time series". *Remote Sens*ing, 9(2), 173-197, 2017.
[18] Shanthini A, Vinodhini G, Chandrasekaran RM. "A taxonomy on impact of label noise and feature noise using machine learning techniques". *Soft Computing,* 23, 8597-8607, 2019.
[19] Kasar M, Bhattacharyya D, Kim TH. "Face recognition using neural network: A review". *International Journal of Security and Its Applications*, 10, 81-100, 2016.
[20] Chandra MA, Bedi SS. "Survey on SVM and their application in image classification". *International Journal of Information Technology*, 13, 1-11, 2018.
[21] Fletcher S, Islam Z. "Decision tree classification with differential privacy: A survey". *ACM Computing Surveys*. 52(4), 1-33, 2019.
[22] Sluban B, Lavrac N. "Relating ensemble diversity and performance: a study in class noise detection". *Neurocomputing*, 160, 120-131, 2015.
[23] Chen X, Kang Q, Zhou M, Wei Z. "A novel under-sampling algorithm based on Iterative-Partitioning Filters for imbalanced classification". *2016 IEEE International Conference on Automation Science and Engineering*, Fort Worth, TX, USA, 21-25 August 2016.
[24] Sáez JA, Luengo J, Stefanowski J, Herrera F. "SMOTE-IPF: addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering". *Information Sciences*, 291(5), 184-203, 2015.
[25] García S, Luengo J, Herrera F. "Tutorial on practical tips of the most influential data preprocessing algorithms in data mining". *Knowledge Based Systems*. 98, 1-29, 2016.
[26] Alcala-Fdez J, Fernández A, Luengo J, Derrac J, Garcia S, Sanchez L, Herrera F. "KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework". *Journal of Multiple-Valued Logic and Soft Computing*, 17, 255-287, 2010.
[27] Yadav A. "A survey on unsupervised clustering algorithm based on k-means clustering". *International Journal of Computer Applications*, 156(8), 6-9, 2017.
[28] Zadedehbalaei A, Bagheri A, Afshar H. "A study on DBSCAN clustering algorithm issues and a survey on its improvements". *Soft Computing Journal*, 6(1), 2-37, 2017.
[29] Nwadiugwu M. "Gene-Based clustering algorithms: Comparison between denclue, fuzzy-C, and BIRCH". *Bioinformatics and Biology Insights,* 14, 1-6, 2020.
[30] Kanagala HK, Jaya Rama Krishnaiah VV. "A comparative study of K-Means, DBSCAN and OPTICS". *2016 International Conference on Computer Communication and Informatics*, Coimbatore, India, 7-9 January 2016.
[31] Schelling B, Plant C. "KMN-removing noise from k-means clustering results". *Big Data Analytics and Knowledge Discovery 2018*, Regensburg, Germany, 3-6 September 2018.
[32] Gan G, Kwok-Po Ng M. "K-means clustering with outlier removal". *Pattern Recognition Letters*, 90, 8-14, 2017.
[33] Cigdem B, Katsageorgiou V, Fisher RB. "Extracting statistically significant behaviour from fish tracking data with and without large dataset cleaning". *IET Computer Vision*, 12(2), 162-170, 2018.
[34] Meeyai S. "Logistic regression with missing data: A comparison of handling methods, and effects of percent Missing Values". *Journal of Traffic and Logistics Engineering*, 4(2), 128-134, 2016.
[35] Ryu S, Kim M, Kim H. "Denoising autoencoder-based missing value imputation for smart meters". *IEEE Access*, 8, 40656-40666, 2020.
[36] Zhang Z. "Missing data imputation: focusing on single imputation". *Annals of Translational Medicine*, 4(1), 9-17, 2016.
[37] Shao X, Wu S, Feng X, Song R. "Categorical missing data imputation approach via sparse representation". *International Journal of Services Technology and Management*, 22, 256-270, 2016.
[38] Chomboon K, Chujai P, Teerarassammee P, Kerdprasop K. "An empirical study of distance metrics for k-Nearest neighbor algorithm". *International Conference on Industrial Application Engineering 2015*, Kitakyushu, Japan, 28-21 March 2015.
[39] Zhongguo Y, Hongqi L, Liping Z, Qiang L, Ali S. "A case based method to predict optimal k value for k-NN algorithm". *Journal of Intelligent & Fuzzy Systems*, 33(1), 55-65, 2017.
[40] de Silva H, Perera AS. "Missing data imputation using Evolutionary k- Nearest neighbor algorithm for gene expression data". *2016 Sixteenth International Conference on Advances in ICT for Emerging Regions*, Negombo, Sri Lanka, 1-3 September 2016.
[41] He Y, Pi D. "Improving KNN method based on reduced relational grade for microarray missing values imputation". *IAENG International Journal of Computer Science*, 43(3), 356-362, 2016.
[42] Lee JY, Styczynski MP. "NS-kNN: a modified k-nearest neighbors approach for imputing metabolomics data". *Metabolomics,* 14(12), 153-165, 2018.
[43] Fletcher S, Islam Z. "Decision tree classification with differential privacy: A survey". *ACM Computing Surveys*. 52(4), 1-33, 2019.
[44] Gavankar SS, Sawarkar SD. "Eager decision tree". *2017 2nd International Conference for Convergence in Technology*, Mumbai, India, 7-9 April 2017.

[45] Khan S, Wimmer H, Powell L. "Open vs. close source decision tree algorithms: comparing performance measures of accuracy, sensitivity and specificity". *2017 Proceedings of the Conference on Information Systems Applied Research*, Austin, Texas, USA, 5-8 November 2017.
[46] Davis D, Rahman M. "Missing value imputation using stratified supervised learning for cardiovascular data". *Journal of Informatics and Data Mining*, 1(2), 1-9, 2016.
[47] Abidin NZ, Ismail AR, Emran N. "Performance analysis of machine learning algorithms for missing value imputation". *International Journal of Advanced Computer Science and Applications*, 9(6), 442-447, 2018.
[48] Kamble VB, Deshmukh SN. "Comparision between accuracy and MSE, RMSE by using proposed method with imputation technique". *Oriental Journal of Computer Science and Technology,* 10, 773-779, 2017.
[49] Raja PS, Thangavel K. "Missing value imputation using unsupervised machine learning techniques". *Soft Computing*, 24, 4361-4392, 2020.
[50] Aljuaid T, Sasi S. "Proper imputation techniques for missing values in data sets". *2016 International Conference on Data Science and Engineering*, Cochin, India, 23-25 August 2016.
[51] Venkatesh B, Anuradha J. "A review of feature selection and its methods". *Cybernetics and Information Technologies*, 19(1), 3-26, 2019.
[52] Urbanowicz RJ, Meeker M, La Cava W, Olson RS, Moore JH. "Relief-based feature selection: Introduction and review". *Journal of Biomedical Informatics*, 85, 189-203, 2018.
[53] Tripathi A, Trivedi SK. "Sentiment analyis of Indian movie review with various feature selection techniques". *2016 IEEE International Conference on Advances in Computer Applications*, Coimbatore, India, 24-24 October 2016.
[54] Liu M, Xu L, Yi J, Huang J. "A feature gene selection method based on ReliefF and PSO". *2018 10th International Conference on Measuring Technology and Mechatronics Automation*, Changsha, China, 10-11 February 2018.
[55] Wosiak A, Zakrzewska D. "Integrating correlation-based feature selection and clustering for improved cardiovascular disease diagnosis". *Complexity*, 2018(1), 1-11, 2018.
[56] Chuanlei Z, Shanwen Z, Jucheng Y, Yancui S, Jia C. "Apple leaf disease identification using genetic algorithm and correlation based feature selection method". *International Journal of Agricultural and Biological Engineering*, 10(2), 74-83, 2017.
[57] Amarnath B, & Balamurugan S. "Review on feature selection techniques and its impact for effective data classification using UCI machine learning repository dataset". *Journal of Engineering Science and Technology*, 11, 1639-1646, 2016.
[58] Uzer M, Yılmaz N, Inan O. "Feature selection method based on artificial bee colony algorithm and support vector machines for medical datasets classification". *The Scientific World Journal*, 2013(11), 1-10, 2013.
[59] Pasyuk A, Semenov E, Tyuhtyaev D. "Feature selection in the classification of network traffic flows". *2019 International Multi-Conference on Industrial Engineering and Modern Technologies*, Vladivostok, Russia, 1-4 October 2019.
[60] Gacav C, Benligiray B, Topal C. "Sequential forward feature selection for facial expression recognition". *2016 24th Signal Processing and Communication Application Conference*, Zonguldak, Turkey, 16-19 May 2016.
[61] Widiyanti E, Endah SN. "Feature selection for music emotion recognition". *2018 2nd International Conference on Informatics and Computational Sciences,* Semarang, Indonesia, 30-31 October 2018.
[62] Yulianti Y, Saifudin A. "Sequential feature selection in customer churn prediction based on naive bayes". *IOP Conference Series: Materials Science and Engineering*, Bandung, Indonesia, 4-9 October 2020.
[63] Wang M, Lu Y, Qin J. "A dynamic MLP-based DDoS attack detection method using feature selection and feedback". *Computers & Security*, 88, 1-14, 2019.
[64] Muthukrishnan R, Rohini R. "LASSO: A feature selection technique in predictive modeling for machine learning". *2016 IEEE International Conference on Advances in Computer Applications*, Coimbatore, India, 24-24 October 2016.
[65] Osman H, Ghafari M, Nierstrasz O. "Automatic feature selection by regularization to improve bug prediction accuracy". *2017 IEEE Workshop on Machine Learning Techniques for Software Quality Evaluation (MaLTeSQuE),* Klagenfurt, Austria, 21-21 February 2017.
[66] Panda D, Ray R, Abdullah AA, Dash SR. "Predictive systems: Role of feature selection in prediction of heart disease". *International Conference on Biomedical Engineering,* Penang island, Malaysia, 26-27 August 2019.
[67] Hart P. "The condensed nearest neighbor rule (Corresp.)". *IEEE Transactions on Information Theory*, 14(3), 515-516, 1968.
[68] Wilson DL. "Asymptotic properties of nearest neighbor rules using edited data". *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-2(3), 408-421, 1972.
[69] Kasemtaweechok C, Suwannik W. "Prototype selection for k-nearest neighbors classification using geometric median". *Proceedings of the Fifth International Conference on Network, Communication and Computing*, Kyoto, Japan, 17-21 December 2016.
[70] García-Pedrajas N, Romero del Castillo JA, Cerruela-García G. "A proposal for local k values for k-nearest neighbor rule". *IEEE Transactions on Neural Networks and Learning Systems*, 28(2), 470-475, 2017.
[71] Song Y, Liang J, Lu J, Zhao X. "An efficient instance selection algorithm for k nearest neighbor regression". *Neurocomputing*, 251, 26-34, 2017.
[72] Pan J, Zhuang Y, Fong S. "The impact of data normalization on stock market prediction: Using SVM and technical indicators". *International Conference on Soft Computing in Data Science*, Kuala Lumpur, Malaysia, 21-22 September 2016.
[73] Singh D, Singh B. "Investigating the impact of data normalization on classification performance". *Applied Soft Computing*, 97, 1-23, 2020.
[74] Pandey A, Jain A. "Comparative analysis of knn algorithm using various normalization techniques". *International Journal of Computer Network and Information Security*, 9, 36-42, 2017.

[75] Eesa A, Arabo W. "A normalization methods for backpropagation: A comparative study*". Science Journal of University of Zakho*, 5(4), 319-323, 2017.

[76] Ali A, Senan N. "The effect of normalization in violence video classification performance". *IOP Conference Series: Materials Science and Engineering*, Melaka, Malaysia, 6-7 May 2017.

[77] Zhang B, Yi Y, Wang H, Yu J. "MIC-TJU at mediaeval violent scenes detection (VSD)". *Multimedia Evaluation Workshop, Barcelona, Spain,* 16-17 October 2014.

[78] Harb H, Makhoul A, Tawbi S, Couturier R. "Comparison of different data aggregation techniques in distributed sensor networks". *IEEE Access*, 5, 4250-4263, 2017.

[79] Morell A, Correa A, Barceló M, Vicario JL. "Data aggregation and principal component analysis in WSNs". *IEEE Transactions on Wireless Communications*, 15(6), 3908-3919, 2016.

[80] Xie Y, Chen X, Zhao J. "Data fault detection for wireless sensor networks using multi-scale PCA method". *2011 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce*, Deng Feng, China, 8-10 August 2011.

[81] Li J, Guo S, Yang Y, He J. "Data aggregation with principal component analysis in big data wireless sensor network". *2016 12th International Conference on Mobile Ad-Hoc and Sensor Networks*, Hefei, China, 16-18 December 2016.

[82] Yu T, Wang X, Shami A. "Recursive principal component analysis-based data outlier detection and sensor data aggregation in IoT systems". *IEEE Internet of Things Journal*, 4(6), 2207-2216, 2017.

[83] Boubiche S, Boubiche DE, Bilami A, Toral-Cruz H. "Big data challenges and data aggregation strategies in wireless sensor networks". *IEEE Access*, 6, 20558-20571, 2018.

[84] Yıldız K, Camurcu Y, Doğan B. "Comparison of dimension reduction techniques on high dimensional datasets". *International Arab Journal of Information Technology*, 15(2), 256-262, 2018.

[85] Choudhari E, Bodhe KD, Mundada SM. "Secure data aggregation in WSN using iterative filtering algorithm". *2017 International Conference on Innovative Mechanisms for Industry Applications,* Bangalore, India, 21-23 February 2017.