Preprocessing Methods and Pipelines of Data Mining: An Overview

Preprint · June 2019

Canchen Li
Zhejiang University

Preprocessing Methods and Pipelines of Data Mining: An Overview

Li Canchen
Department of Informatics
Technical University of Munich
Canchen.Li@in.tum.de

***Abstract*—Data mining is about obtaining new knowledge from existing datasets. However, the data in the existing datasets can be scattered, noisy, and even incomplete. Although lots of effort is spent on developing or fine-tuning data mining models to make them more robust to the noise of the input data, their qualities still strongly depend on the quality of it. The article starts with an overview of the data mining pipeline, where the procedures in a data mining task are briefly introduced. Then an overview of the data preprocessing techniques which are categorized as the data cleaning, data transformation and data preprocessing is given. Detailed preprocessing methods, as well as their influenced on the data mining models, are covered in this article.**

***Index Terms*—Data Mining, Data Preprocessing, Data Mining Pipeline**

I. Introduction

DATA mining is a knowledge obtaining process: it gets data from various data sources and finally transforms the data into knowledge, thus provides insight to its application field. Data mining pipeline is a typical example of the end-to-end data mining system: they are an integration of all data mining procedures and deliver the knowledge directly from data source to human.

The purpose of data preprocessing is making the data easier for data mining models to tackle. The quality of data can have a significant influence on data mining models. It is considered that the data and features have already set the upper bound of the knowledge that can be obtained, and the data mining models are just about approximating the upper bound. Various preprocessing techniques are invented to make the data meet the input requirements of the model, improve the relevance of the prediction target, and make the optimization step of the model easier.

It is common that raw data obtained from the natural world is badly shaped. The problems include the appearance of missing values (e.g., a patient did not go through all the tests), duplications (e.g., annual income and monthly income), outlier values (e.g., age is -1) as well as contradictions (e.g., gender is male and is pregnant) in the dataset. Although the existing preprocessing techniques would not guarantee to solve all these problems, they could at least correct some of them and improve the performance of the models.

The data type and distribution of data are usually transformed before being sent to data mining models. The purpose of data transformation includes making the data meets the input requirement of the models, removing the noise of data, and making the distribution of data more suitable for applying optimization algorithms in the model training step.

The input for data mining models can be huge: they may have too many dimensions or of massive amount, which would make it difficult for the data mining model to train or cause troubles while transferring and storing the data. Data reduction techniques can reduce the problem by applying reduction on dimensions (known as dimensional reduction) or amounts of data (known as instance selection and sampling).

To implement preprocessing to data, Python and R are among the most popular tools. With bulks of packages such as scikit-learn [1] and PreProcess [2], most of the preprocessing algorithms covered in this paper can be implemented even without consideration of its details.

In the following section, the data mining pipeline and the primary procedures in the data mining pipeline will be introduced. From Section 2 on, we will focus on the steps in the data preprocessing work: Section 3 will introduce the techniques used in data cleaning, while Section 4 will cover the data transformation techniques. In the last section, data reduction techniques will be discussed.

II. Data Mining Pipeline

The data mining pipeline is an integration of all procedures in a data mining task. While most of the data already exists in a data base, a data warehouse, or other types of data source [3], various steps should be taken in order to make them easier for a human to understand. An illustration of the data mining pipeline is given as in Fig. 1. Generally speaking, the key procedures include *obtaining*, *scrubbing*, *exploring*, *modelling*, *interpreting*. These procedures are known as "OSEMN"[1]. However, note that the pipeline is not a linear process in the real world, but a successive and long-lasting task. Methods in scrubbing and modeling procedures have to be tested and refined, the obtaining procedure may have to be adapted for different kinds of data sources, and the visualization and interpretation of the data may have to be adjusted for their audience, thus meet the audience's demand. In the rest of this section, details of these procedures will be discussed.

A. Obtaining

Obtaining the data is the most fundamental step in data mining since it is the data itself that decides what knowledge

[1] http://www.dataists.com/2010/09/a-taxonomy-of-data-science/

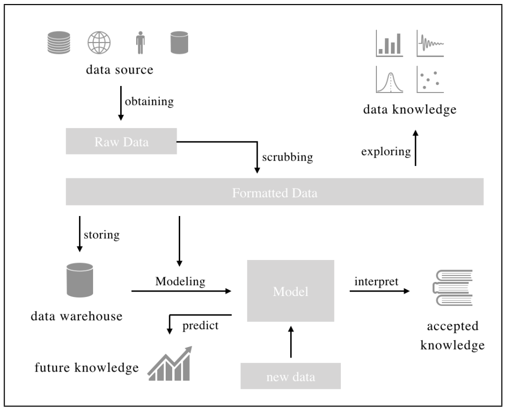

Fig. 1. An illustration of data mining pipeline.

it may contain. Data base and data warehouse are among the primary sources of data, where the structured data can be fetched with query languages, usually SQL. The data warehouse is specially designed for organizing, understand, and making use of the data [3]: they are usually separated system from the operational database, having time-variant structure as well as structures that makes the subsequent analysis work easy, and most importantly, nonvolatile.

The obtained data can be archived as files and directly used for the subsequent procedures. They may also be reformatted and stored in a data base or a data warehouse, prepared for the data mining tasks in the future.

In the past, as well as in most common cases now, we regard the data obtaining step as the process for obtaining a dataset, regardless of how we obtained them. However, nowadays, new data is generating at an extreme speed: there are tons of data being created every second of every day. Some services, such as the public opinion monitoring and the recommendation system, do need the newly generated data: they have strong demand for being on time. In these circumstances, the concept of ”stream” is comparatively more important than a dataset. A stream is a real-time representation of data. Under this concept, models and algorithms that can run online are developed [4]. For the stream mining tasks, the goal of data obtaining is no longer obtaining a dataset, but a real-time input source.

B. Scrubbing

Scrubbing is about the cleaning and preprocessing of the data, aiming to make the data have a unified format and easy to be modeled. As for the detailed concepts and techniques in data scrubbing, the readers could refer to the following sections of this paper, since most of them will be covered in the overview of data preprocessing.

C. Exploring

Before modeling the data, people may want to get to know the underlying distribution of the data, the correlation between variables, and their correlation with the labels. Assumptions can be made in this step. For instance, people may assume smoking is highly correlated with lung cancer. These assumptions are important because they would provide indications to other procedures in a data mining task, including helping to choose a suitable model, and helping to justify your work in the interpreting of data.

Tools for exploring the data and verifying the assumptions are usually statistical analysis and data visualization. The statistical analysis gives us theoretical probabilities, known as significance level, of our assumption being incorrect, while data visualization tools, such as ggplot [5] and D3 [6], give us the impressions about the distribution of the data, help people to verify their assumption conceptually. Also, new patterns that are ignored in the assumption step might be found in the visualization step.

D. Modeling

With underlying patterns existed in the data source, modeling makes it possible to represent the pattern explicitly with the data mining models. For a data mining task, modeling would usually split the data into the training set and test set thus could score the accuracy of the model on a relatively ”new” dataset. If the model contains hyperparameters, such as the parameter k in a K-Nearest Neighbor (KNN) model, a cross validation set will be created for obtaining the best set of the hyperparameter.

For most of the data mining models, loss functions are defined. Generally, a loss function will have lower value if the model performs well. Besides, it usually has special features such as convexity, which makes the gradient-based optimizing algorithm performs better. With trainable parameters, a model's training step is about adjusting its parameter so that it gains lower loss on its training data. The specific definition of loss function depends on the model itself and the task. Mean squared error ($\sum_{i=1}^{n}(\hat{y}_i - y_i^2)$) for regression task and cross entropy ($-\sum_{i=1}^{n}[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)]$) for classification task are frequently accepted loss functions.

There are too many kinds of data mining models existed; their tasks include clustering, classification, and regression. The complexity of the models also varies: simple models such as linear regression only have a few parameters, a small amount of data will make the training step converge, while complex models such as AlexNet have millions of parameters [7], and their training also require huge dataset. However, complex does not mean better: the model should be decided according to the predicting target of the task, the dataset size, the data type, etc. Sometimes, it is necessary to run different models over one dataset and find the most suitable data mining model.

E. Interpreting

While previous steps are generally pure science, the interpreting step is more humanistic. Knowledge can be extracted from the input data, but it takes extra effort to convince people to accept this knowledge. Although complicated statistics and models make the work looks more professional, for laymen, graphs, tables and well explained accuracy make it easier to understand and accept. Besides, social skills such as story telling and emotion quotient are also important: this procedure is about the human, not the data.

III. Data Cleaning

The data obtained from the natural world is usually badly shaped. Some of the problems, such as outlier, may affect the data mining model and produce a biased result. For instance, the outlier may affect the K-Means clustering algorithm by substantially distort the distribution of data [8]. Other problems, if not handled, will make it impossible for the data to be analyzed by models, such as the Not a Number (nan) values in a data vector. Data cleaning techniques, including missing value handling and outlier detection, were issued to tackle the problem. They make the gathered data suitable as the input of the model.

A. Missing Values Handling

Missing values is a typical kind of data incompleteness of dataset. Most of the data mining models would not tolerate the missing values of its input data: these values can not be used for comparison, not available for categorizing, and can not be operated with arithmetic algorithms. Thus, it is necessary to handle the missing values before pushing the dataset to data mining models. The easiest way to deal with missing value is to drop the entire sample. This method is effective if the proportion of missing value in a dataset is not significant, however, if the number of missing values is not suitable for ignoring, or the percentage of missing value for each attribute is different [3], dropping the samples with missing value would reduce the amount of dataset dramatically, the information contained in the dropped samples is not made use of. Another way to deal with missing values is by filling them, and there are varies methods for finding the suitable value to fill the missing value, some of them are listed as follows.

1) Use special value to represent missing: Sometimes the missing value itself has some meaning. For instance, in a patient's medical report, the missing value for uric acid means the patient did not go through the renal function test. Thus, using a certain value such as -1 makes sense, for they can be operated like normal values while having special meaning in the dataset.

2) Use attribute statistics to fill: Statistics such as mean, median, or mode can be obtained from non-missing values in the missing value's corresponding attribute. It is said that for a skewed dataset, the median would be a better choice [3]. However, this technique does not take the sample's other non-missing attributes into account.

3) Predicting the value with known attributes: If we assume, there exists a correlation between attributes, filling the missing value can be modeled as a prediction problem: predicting the value with the non-missing attributes with other samples as training data. The predicting methods includes regression algorithms, decision trees [3], and K-Means [9].

4) Assigning all possible values: For categorical attributes, given an example E with m possible values for its missing value, then E can be replaced with m new examples $E_1, E_2, \ldots, E_m$. This missing value filling technique assumes the missing attribute does not matter for the example. Thus the value can be anyone in its domain [10].

Comparison of different missing value filling techniques is done in [11]. In the work, different missing value filling techniques are tested on ten datasets for running simple and extended classification methods. The conclusion shows C4.5 decision tree method performs the best, ignoring the samples with missing values and assigning all possible values also performed well, and filling the values with mode perform the worst. However, the performance of missing value filling techniques may differ because of the feature of the dataset. As a result, most of the techniques are worth trying for a data mining task.

B. Outlier Detection

Outlier refers to the data sample that has a massive distance to most of the other samples. Although the rare case does not necessarily mean wrong (e.g., age = 150), most outliers are caused by measurement error or wrong recording, thus ignoring a rarely appearing case would not harm a lot. Although some of the models are robust against outliers, outlier detection is still recommended in data preprocessing work.

Statistics-based outlier detection algorithms are among the most commonly used algorithms, which assume an underlying distribution of the data [12] and regard the data examples which corresponding probability density lower than a certain threshold as the outliers. As the underlying distribution is unknown for most cases, the normal distribution is a good substitute, and its parameter could be estimated by the mean value and standard deviation of the data. The Mahalanobis distance [12], as in (1), is a scale-irrelevant distance between two data samples. The outlier can be decided by comparing the Mahalanobis distance between each sample and the mean value of all samples. Box-plot, as another kind of statistics-based outlier detection technique, can give the graphical representation of outlier by plotting the lower quartile and upper quartile along with the median [13].

$$D_M(x,y) = \sqrt{(x-y)^T \Sigma^{-1} (x-y)} \tag{1}$$

Without making any assumption of the distribution of the data, the distance-based outlier detection algorithm can detect the outlier by analyzing the distance between every two samples, thus determine the outliers. Simple distance-based outlier detection algorithms are not suitable for a large dataset, since for n samples with m dimensions, their complexity is usually $O(n^2m)$ [12], and each computation requires scanning all the samples. However, an extended cell-based outlier detection algorithm is developed in [14], which guarantee linear complexity over the dataset volume and no more than three dataset scans. The experiment shows this algorithm is the best for the dataset with dimension less than 4.

Sometimes, with consideration of temporal and spatial locality, an outlier may not be a separate point, but a small cluster. Cluster-based outlier detection algorithms consider clusters with small size as outlier clusters and clean the dataset by removing the whole cluster [15] [16].

IV. Data Transformation

The representation of data in different attributes varies: some are categorical, while some are numerical. For categorical values, they can be nominal, binary or ordinal [3], and for numerical data, they can also have different statistical features including mean values and standard deviations. However, not all kinds of data meet the requirement of data mining models. Also, the difference among data attributes may bring troubles for the subsequent optimization work of data mining models. Data transformation is about modifying the representation of data so that they are qualified to be the input for data mining models, as well as making the optimization algorithm of the data mining model easier to take effect.

A. Numeralization

Categorical values widely exists in the natural world, some of the operations, such as calculating the entropy between groups, can be done directly over categorical data. However, most operations are not applicable to categorical data. Thus, categorical data is supposed to be encoded into numerical data, making it meets the requirements of the models. The following encoding techniques are adopted for numeralization.

- *One-Hot encoding*: Regard each possible value of the categorical data as a single dimension, and use 1 for the dimension which the sample belongs to the category, otherwise 0.
- *Sequential encoding*: For each possible value of the categorical data, assign it with a unique and numerical index. This is implemented as a kind of word encoding, as in [17].
- *Customized encoding*: Customized encoding is based on rules designed for a certain task. For instance, word2vec [18] is an encoding that can turn a word into a 300 dimensional vector, with consideration of the word's meaning.

Generally, one-hot encoding is suitable for categorical data with fewer possible values; if there are too many possible values, such as English words, the encoded dataset would be huge and sparse. Sequential encoding would not produce huge output, but the encoded data is not as easy to separate as one-hot encoded data. Customized encoding, if carefully designed, usually perform well over a certain kind of task, but for other tasks, the encoding should be redesigned, and its design can take lots of efforts.

B. Discretization

Discretization of data is applied sometimes to meet the requirement of input of models, such as Naive Bayes which require its input to be countable [19]. Also, it can smooth the noise. Discretization of data does not necessarily make the data categorical, but make the continuous values countable. The discretization of data can be achieved with unsupervised learning methods such as putting data into equal-width or equal-frequency slots, known as *binning*, or clustering. Some supervised learning methods such as *decision tree* can also be used for discretization of data [3].

C. Normalization

Since different attribute usually adopts a different unit system, their mean value and standard deviation are usually not identical. However, the numerical difference would make some of the attributes look more ”important”, while others are not [3]. This impression could cause trouble for some models; one of the typical ones is KNN: larger value would strongly affect the distance comparison, making the model mainly consider attributes tend to have larger numerical values. Besides, for neuron network models, the different unit system will also have a negative influence on gradient descent optimizing methods, forcing it to adopt a smaller learning rate. To tackle the problem mentioned above varies of normalization methods are issued, some of them are listed as follows.

1) Min-max normalization: Min-max normalization is used for mapping the attribute from its range $[lb, ub]$ to another range $[lb_{new}, ub_{new}]$; the target range is usually $[0, 1]$ or $[-1, 1]$ [20]. For a sample with value v, the normalized value v' is given as in (2).

$$v' = \frac{v - lb}{ub - lb}(ub_{new} - lb_{new}) + lb_{new} \tag{2}$$

2) Z-score normalization: If the underlying range of an attribute is unknown or outlier exists, min-max normalization is not feasible or could be strongly affected [20]. Another normalization approach is to transform the data so that it would have 0 as mean and 1 as standard deviation. Given the mean μ and standard deviation σ of the attribute, the transformation is represented as in (3).

$$v' = \frac{v - \mu}{\sigma} \tag{3}$$

Note that if μ and σ are unknown, they can be substituted with the sample mean and standard deviation.

3) Decimal scaling normalization: An easier way to implement the normalization is to shift the floating point of the data so that each value in an attribute would have an absolute value less than 1, the transformation is given as in (4).

$$v' = \frac{v}{10^j} \tag{4}$$

For some cases, different attributes have an identical or similar unit system, such as the preprocessing of RGB-colored imaged. In these cases, normalization is not necessary. However, if this is not guaranteed, normalization is still recommended for all data mining tasks.

D. Numerical Transformations

The transformation over dataset can help to obtain additional attributes. These features obtained by transformation could be unimportant for some data mining models, such as neural network, which have superior fitting potential. However, for relatively simpler models with fewer parameters, linear regression, for example, the transformed features do help the model to get better performance(as in Fig. 2), for they could provide additional indication of the relationship between

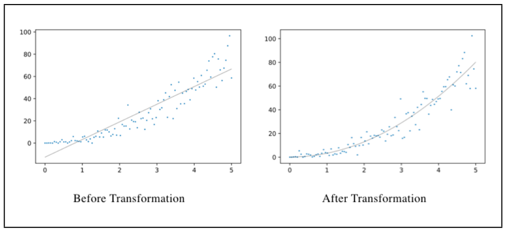

Fig. 2. The effect of box-cox transformation in linear regression. The feature and label are quadratically related.

attributes. The transformation would also be essential for scientific discoveries and machine controls [21].

Generally, given attributes set $\{a_1, a_2, \ldots, a_p\}$, the numerical transformation can be represented as in (5). Theoretically, f can be any function, however, since the input data is finite, f can take polynomial forms [21].

$$x' = f(a_1, a_2, \ldots, a_p) \tag{5}$$

The commonly used representation of f includes *polynomial based transformation*, *approximation based transformation*, *rank transformation* and *box-cox transformation* [20]. The parameters in the transformation formula could be obtained by subjective definition (for situations where people know the relationship between attributes and labels well), by brute search [20] or by applying maximum likelihood method.

V. Data Reduction

The amount of data in a data warehouse or a dataset can be huge, causing difficulties for data storage and processing when working on a data mining task, while not every model needs a huge amount of data to train. On the other hand, although the data may have lots of attributes, there could be unrelated features as well as the interdependence between features [22]. Data reduction is the technique that helps to reduce the amount or dimension, or both, of a dataset, thus making the model's learning process more efficient as well as helping the model to obtain better performance, including preventing overfitting problem, and fix the skewed data distribution.

A. Dimensional reduction

The dimension reduction technique is about reducing the dimensionality of data samples, thus reduce the total size of the data. As the number of attributes is reduced for a sample, there is less information contained in it. A good dimensional reduction algorithm will keep more general information: this could make it more difficult for models to become overfitted.

Some dimensional reduction techniques pose a dimensional reduction transformation over a dataset, generating new data samples, which have a fewer number of attributes than before. The transformations have different criteria. Principal component analysis, known as PCA, could reduce the dimension of data while keeping the maximum variance of data [23]. This is achieved by multiplying matrix $A = (a_1, a_2, \ldots, a_p)^T$ to the dataset X and keep the top k dimensions (a_i stands for the normalized eigenvector corresponds to the i^{th} greatest eigenvalue of the covariance matrix of the dataset). By contrast, linear discriminant analysis (LDA), is meant to maximize the component axes for class-separation. The implementation of LDA is similar to PCA; the only difference is it replaces the covariance matrix with the scatter matrix of samples. A graphical illustration of the difference between PCA and LDA is given as Fig. 3. In comparatively more situations, LDA outperforms PCA. PCA may outperform LDA when the data amount is small, or the data is nonuniformly sampled [24]. Other dimensional reduction algorithms include factor analysis (assuming a lower dimensional underlying distribution), projection pursuit(measuring the aspect of non-Gaussianity) [25], and wavelet transform [3].

Feature selection is another dimensional reduction technique: it is about removing irrelevant or correlated attributes from the dataset while keeping the other relatively independent attributes untouched. Feature selection is more than simply selecting the feature that has greater relevance with the variable to predict, the relationship between attributes are also supposed to be taken into consideration: the goal is to find a sufficiently good subset of features to predict [22]. Feature selection methods can be divided into three types, as follows [26].

- *filter*: Directly select the feature based on attribute level criteria, including information gain, correlation score, or chi-square test. The filter method does not take the data mining model into consideration.
- *wrapper*: Use techniques to search through the potential subsets, according to their performance on the data mining model. Greedy strategies, including forward selection and backward elimination [26], are issued in order to reduce the time consumption.
- *embedded*: Embed the feature selection into the data mining model. Usually, the weight over different attributes would act as feature selection. A typical example is the regularization term of the loss function in linear regression, known as Lasso regression (for L1 regularization) and ridge regression (for L2 regularization).

B. Instance selection and sampling

Both instance selection and sampling are about achieving the reduction of data by reducing the amount of data, seeking the chance to train the model with minimum performance loss [20], while based on different criteria for selecting (or dropping) instance.

Most instance selection algorithm is based on fine-tuning classification models. To help the model make better decision, *condensation algorithm* and *edition algorithm* are issued [20]. The condensation removes the samples lie in the relative center area of the class, assuming they do not contribute much in classification. *Condensed nearest neighbor* [27], for instance, select instance by adding all the samples that cause a mistake to a K-Nearest Neighbor classifier. Edition algorithm removes the samples close to boundary, hoping to give the classifier a smoother decision boundary. Related algorithms include a clustering-based algorithm to select the center of clusters [28],

Compared with instance selection methods, sampling is a faster and easier way to reduce the number of instances,

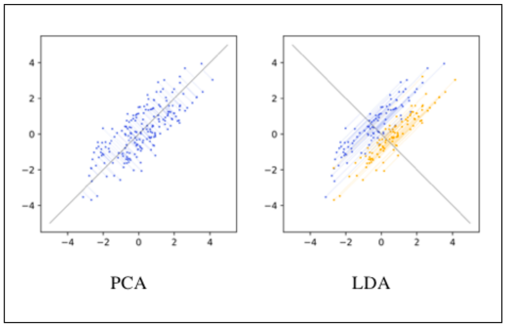

Fig. 3. A comparison of reduction result with PCA and LDA.

since almost no complex selection algorithm is required for sampling methods: they only focus on reducing the amount of the data samples. The easiest sampling technique is *random sampling*, which collect a certain amount or portion of samples from the dataset randomly. For skewed datasets, *stratified sampling* [22] is more adapted, since it takes the appearance frequency of labels from different classes into account and assigns a different probability of data with different labels being chosen, thus makes the sampled dataset more balanced.

VI. Summary and Outlook

Data mining, as a technique to discover additional information from a dataset, can be integrated as a pipeline, in which obtaining, scrubbing, exploring, modeling, and interpreting are the key steps. The purpose of the data mining pipeline is to tackle realistic problems, including reviewing the past and predicting the future. The specific technique used in each step should be selected with care to give the best performance to the pipeline.

The success of a data mining model depends on the proper data preprocessing work. The unpreprocessed data can be of unsuitable format for model input, causing instability for the optimization algorithm of the model, having a great impact on the model's performance because of its noise and outliers, and causing performance problems on the model's training process. With careful selection of preprocessing steps, these problems can be reduced or avoided.

Data type transformation techniques as well as missing value handling techniques makes it possible for models for processing different types of data. By applying normalization, the unit system of different attributes would be more unified, reducing the probability of an optimization algorithm to miss the global minimum. For simpler models, numerical transformation can provide richer features to the model, thus enhance the model's ability to discover more underlying relationships between features and labels. For the overfitting problems of the model, dimensional reduction techniques help model find the more general information about samples instead of the too detailed features by reducing the dimension of feature, thus remove some unimportant information. And for the performance of the model training, both dimensional reduction techniques and instance selection techniques would improve the training performance by reducing the total amount of data.

References

[1] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[2] K. R. Coombes, *PreProcess: Basic Functions for Pre-Processing Microarrays*, 2019, r package version 3.1.7. [Online]. Available: https://CRAN.R-project.org/package=PreProcess

[3] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2011.

[4] M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy, "Mining data streams: a review," *ACM Sigmod Record*, vol. 34, no. 2, pp. 18–26, 2005.

[5] H. Wickham, *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. [Online]. Available: http://ggplot2.org

[6] M. Bostock, V. Ogievetsky, and J. Heer, "D^3 data-driven documents," *IEEE transactions on visualization and computer graphics*, vol. 17, no. 12, pp. 2301–2309, 2011.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[8] T. Velmurugan and T. Santhanam, "Computational complexity between k-means and k-medoids clustering algorithms for normal and uniform distributions of data points," *Journal of computer science*, vol. 6, no. 3, p. 363, 2010.

[9] B. M. Patil, R. C. Joshi, and D. Toshniwal, "Missing value imputation based on k-mean clustering with weighted distance," in *International Conference on Contemporary Computing*. Springer, 2010, pp. 600–609.

[10] J. W. Grzymala-Busse, "On the unknown attribute values in learning from examples," in *International Symposium on Methodologies for Intelligent Systems*. Springer, 1991, pp. 368–377.

[11] J. W. Grzymala-Busse and M. Hu, "A comparison of several approaches to missing attribute values in data mining," in *International Conference on Rough Sets and Current Trends in Computing*. Springer, 2000, pp. 378–385.

[12] M. Kantardzic, *Data mining: concepts, models, methods, and algorithms*. John Wiley & Sons, 2011.

[13] S. Walfish, "A review of statistical outlier methods," *Pharmaceutical technology*, vol. 30, no. 11, p. 82, 2006.

[14] E. M. Knorr, R. T. Ng, and V. Tucakov, "Distance-based outliers: algorithms and applications," *The VLDB JournalThe International Journal on Very Large Data Bases*, vol. 8, no. 3-4, pp. 237–253, 2000.

[15] I. Ben-Gal, "Outlier detection," in *Data mining and knowledge discovery handbook*. Springer, 2005, pp. 131–146.

[16] L. Duan, L. Xu, Y. Liu, and J. Lee, "Cluster-based outlier detection," *Annals of Operations Research*, vol. 168, no. 1, pp. 151–168, 2009.

[17] S. Angelidis and M. Lapata, "Multiple instance learning networks for fine-grained sentiment analysis," *Transactions of the Association of Computational Linguistics*, vol. 6, pp. 17–31, 2018.

[18] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[19] I. Rish *et al.*, "An empirical study of the naive bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22, 2001, pp. 41–46.

[20] S. García, J. Luengo, and F. Herrera, *Data preprocessing in data mining*. Springer, 2015.

[21] T. Y. Lin, "Attribute transformations for data mining i: Theoretical explorations," *International journal of intelligent systems*, vol. 17, no. 2, pp. 213–222, 2002.

[22] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, "Data preprocessing for supervised leaning," *International Journal of Computer Science*, vol. 1, no. 2, pp. 111–117, 2006.

[23] I. Jolliffe, *Principal component analysis*. Springer, 2011.

[24] A. M. Martínez and A. C. Kak, "Pca versus lda," *IEEE transactions on pattern analysis and machine intelligence*, vol. 23, no. 2, pp. 228–233, 2001.

[25] I. K. Fodor, "A survey of dimension reduction techniques," Lawrence Livermore National Lab., CA (US), Tech. Rep., 2002.

[26] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of machine learning research*, vol. 3, no. Mar, pp. 1157–1182, 2003.

[27] P. Hart, "The condensed nearest neighbor rule (corresp.)," *IEEE transactions on information theory*, vol. 14, no. 3, pp. 515–516, 1968.

[28] A. Lumini and L. Nanni, "A clustering method for automatic biometric template selection," *Pattern Recognition*, vol. 39, no. 3, pp. 495–497, 2006.