MEASURE ENERGY CONSUMPTION

Abstract

By the developing technology, the traditional power system has been modernized by advanced metering and communication infrastructures. Thus, a structure with high monitorability and interaction with users has been developed. In modern power systems, applications such as short-term demand forecasting, demand management or dynamic tariffs are actively used. The success of all these applications is closely related to mastering the opportunities provided by the modern power system and using these opportunities effectively. Electrical energy consumption profile clustering is a widely applied and highly functional approach that can be evaluated in this context. It stands out as a practical method that allows obtaining representative load profiles specific to each cluster by determining similar consumption behaviors from big consumption data.

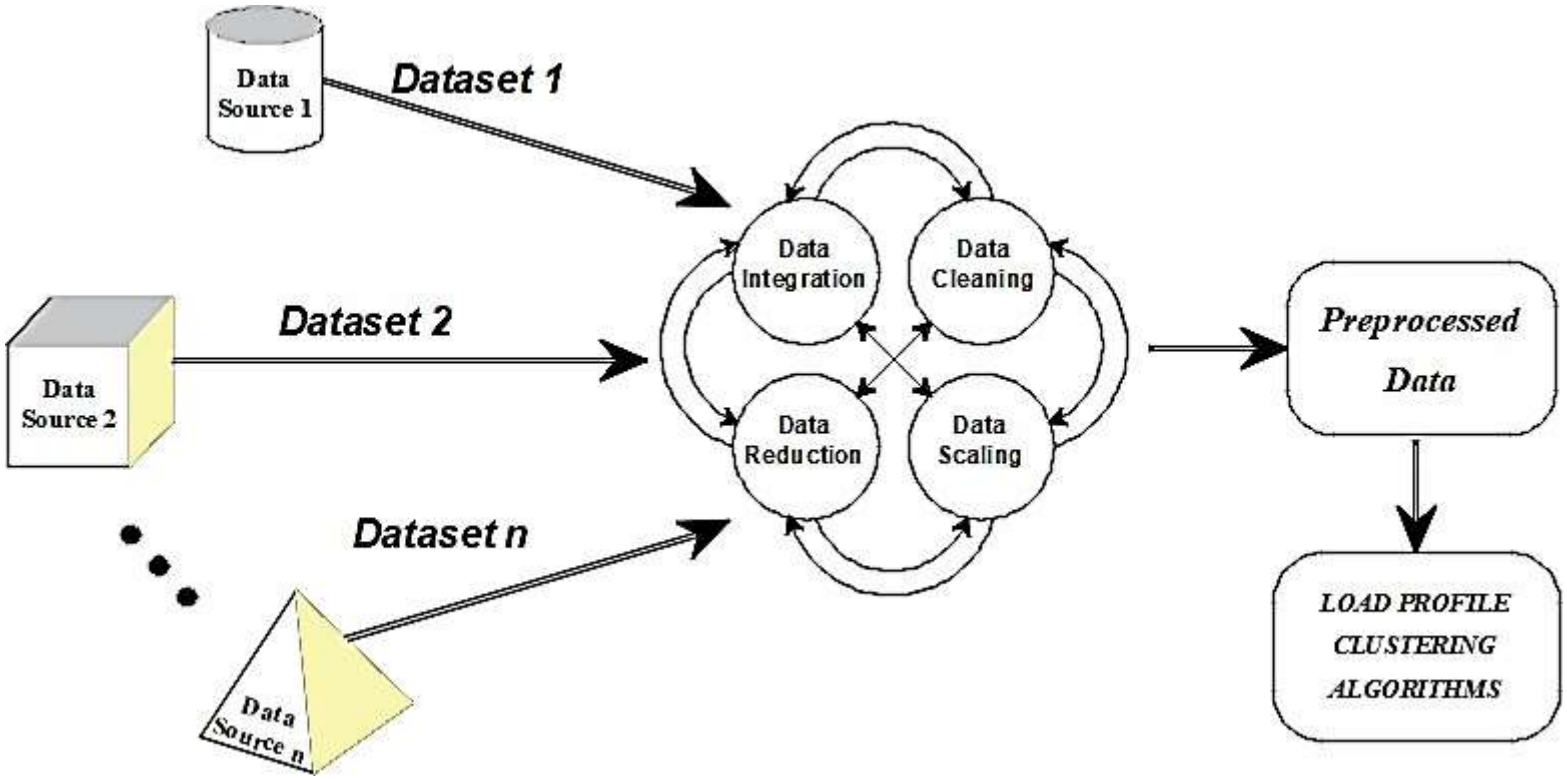
Introduction

- The heating, ventilation and air conditioning (HVAC) systems account for 40%-60% of energy consumption in buildings Optimal operation control for HVAC systems can help improve the energy performance of the HVAC systems dramatically. Accurate short-term energy consumption prediction is the essential prerequisite for developing optimal control. With the development of computer science and the increasing accessibility to large amounts of building operation data, data-driven models are widely used to develop energy consumption prediction models.

Data preprocessing

- By the developments on the measurement and communication infrastructures in power systems, it has become possible to collect data from more points and with higher resolutions compare to the past. Increasing data volume, on the one hand, increases the quality of the information possessed, on the other hand, it has made the processing of data more complicated. With the increase in data volume, the size and variety of data quality problems has also increased. The success of data analysis is closely related to data quality. In order to obtain consistent results, missing or outlier data must be determined and removed from the data sets, and the data should be formatted in accordance with the study. All processes applied for this purpose are called data preprocessing. The data preprocessing is examined under four main headings.

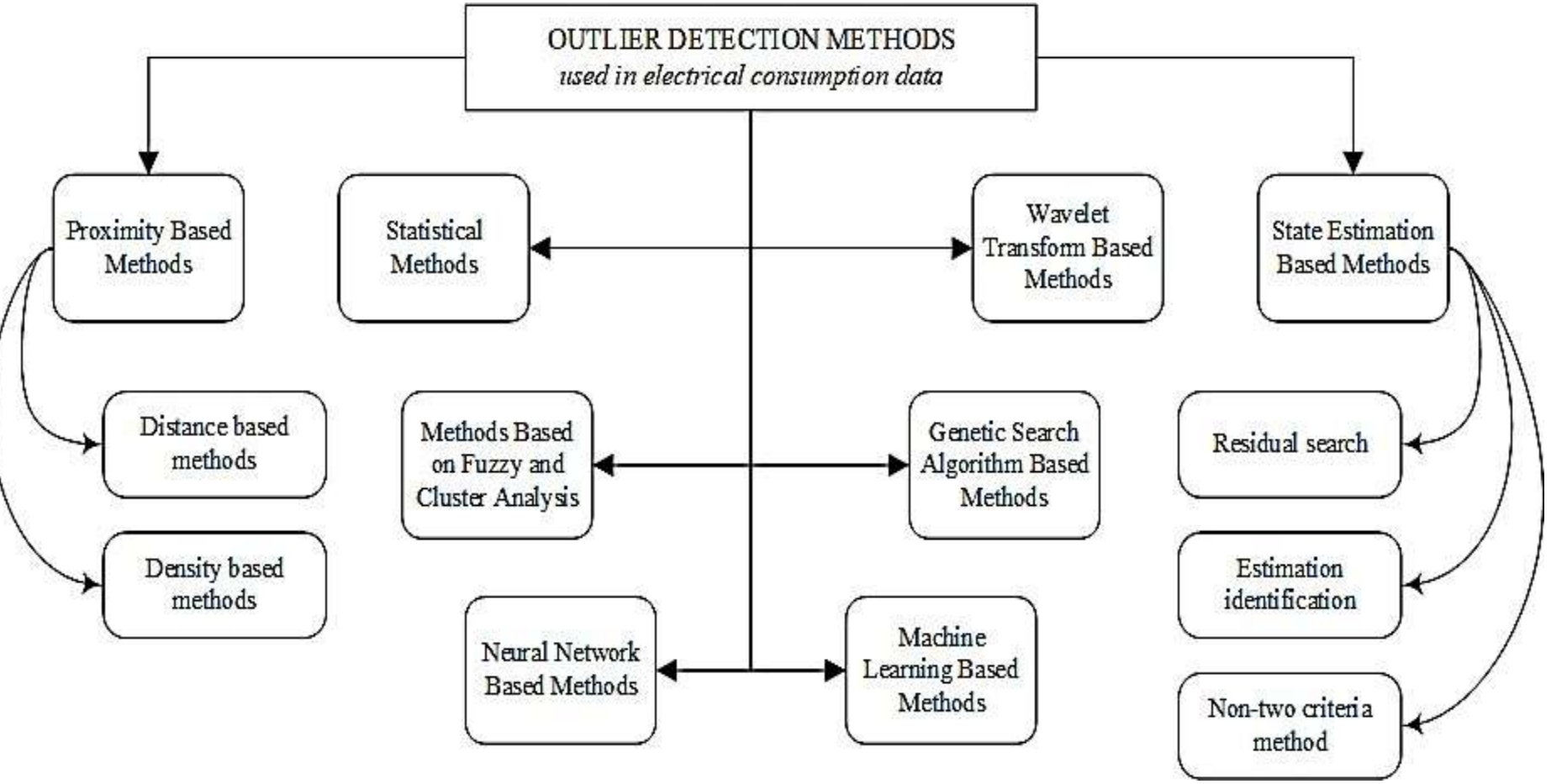
Step


Data Source 1
Dataset 1
Data Source 2
Dataset 2
Dataset n
Data Source n
Data Integration
Data Cleaning
Data Reduction
Data Scaling
Preprocessed Data
LOAD PROFILE CLUSTERING ALGORITHMS

Data integration

- In general, electrical energy consumption clustering studies are based on consumption data only. However, in some studies, various data affecting electricity consumption can also be included in the analysis. In such multivariate studies, different data sets should be combined and analyzes should be performed on a single data set.
- Different data storage mechanisms are used in data storage in order to use data space efficiently and to increase data speed. For this reason, data sets in different databases could be in different data architectures and encrypted. In addition, different formats can be preferred while storing data. In order to create a single usable data set from various databases created independently from each other, it is necessary to eliminate the differences between the data sets and bring them together under a standard structure. This process is called data integration].
- Apart from the structural differences, the content of the data sets to be combined may also differ. Electricity consumption data are time series type data. The difference in the time intervals and resolutions of suchdata sets may cause dimensional incompatibility. On the other hand, not all of the features in a dataset content may be needed. The new data set to be created by selecting only the required features may increase the speed and simplicity of the analysis
- In order to prevent such problems, the data sets can be treated with the help of other pre-processing steps to made ready for integration. It should be noted that there is no hierarchical order between the data preprocessing steps. In case of need, any of preprocessing step can be used over and over again.

Data detection



outlier data and missing data

In its most general definition, it is the values that are far from the general data distribution and are statistically inconsistent with other data. Power system transients or malfunctioning in measurement and communication infrastructure may cause outliers. For example, a value of 300kWh in the hourly energy consumption data of a facility with an installed power of 100kW is an outlier.

Missing data are empty or meaningless sections in the data set as the result of problems in the phase of measurement, transfer, or storage processes.
The first step of data cleaning preprocessing is bad data (outlier, noisy data, or missing data) detection. Noisy and missing data can be detected simply but outlier detection is complicated.

Conclusion

Data science is a very young and rapidly developing field. At the same time, it has a very wide usage area consisting of different disciplines. For these reasons, there is a lot of confusion in both the terminology and the definition and classification of the components. In this study, electricity consumption profile clustering analysis, which is used as the basis of many applications such as short-term demand forecasting, demand management and dynamic tariff planning, has been focused on.

In the first part of the study, data pre-processing steps have been discussed and their applications on electrical energy consumption data has been examined. Studies in the literature have been reviewed and the ones that contain details about the data preprocessing have been taken into account. The data preprocessing steps applied to the electrical energy consumption data and the details of the methods used in the implementation of these steps are presented in tables.

Progarm

```
import java.lang.management.ManagementFactory;
import com.sun.management.OperatingSystemMXBean;

public class EnergyConsumptionMonitor {
    public static void main(String[] args) {
        OperatingSystemMXBean osMBean = (OperatingSystemMXBean)
ManagementFactory.getOperatingSystemMXBean();
        double cpuUsage = osMBean.getSystemCpuLoad() * 100;
        System.out.println("CPU Usage: " + cpuUsage + "%");
    }
}
```

THANK YOU

By

N.Sangeetha