A MapReduce Based k-NN Joins Probabilistic Classifier

Georgios Chatzigeorgakidis*, Sophia Karagiorgou†, Spiros Athanasiou† and Spiros Skiadopoulos*
*University of Peloponnese, Department of Informatics and Telecommunications
Tripolis, Greece
Email: chgeorgakidis@uop.gr
spiros@uop.gr
†R.C. ATHENA, Institute for the Management of Information Systems
Athens, Greece
Email: karagior@imis.athena-innovation.gr
spathan@imis.athena-innovation.gr

***Abstract*—Water management field has concentrated great interest, with the potential to affect the long term well-being, the societal economy and security. In parallel, it imposes specific research challenges which have not been already met, due to the lack of fine-grained data. Knowledge extraction and decision making for efficient management in the energy field has attracted a lot of interest in Big Data research. However, the water domain is strikingly absent, with minimal focused work on data exploitation and useful information extraction. The goal of this work is to discover persistent and meaningful knowledge from water consumption data and provide efficient and scalable big data management and analysis services. We propose a novel methodology which exploits machine learning techniques and introduces a robust probabilistic classifier which is able to operate on data of arbitrary dimensionality and of huge volume. It also provides added value services and new operation models for the water management domain, inducing sustainable behavioural changes for consumers, which can further raise social awareness. It does so through a new k-Nearest Neighbour based algorithm, developed in a parallel and distributed environment, which operates over Big Data and discovers useful knowledge about consumption classes and other water related attitudinal properties. A detailed experimental evaluation assesses the effectiveness and efficiency of the algorithm on prediction precision along with the provision of analytics. The results show that this method is prosperous and provides accurate and interesting results that allow us to identify useful characteristics, not only for the households, but also for the water utilities.**

***Keywords*-big data; mapreduce; classification; forecasting; data mining;**

I. Introduction

Water is one of the most valuable resources for mankind. Experts estimate that water will be the most useful good in the decades to come, and combined with climate change, a potential cause for geopolitical tensions and conflicts. Drinkable water is only a small percentage of existing water bodies and the significance of water management is greatly acknowledged by various local and national policies. To determine how water demand is formulated, identify the factors that influence it, and forecast future demand, the availability, collation, and analysis of water consumption data is considerably required.

In the energy domain, huge amounts of data are generated and feed many applications regarding billing, grid and demand management in a fine-grained or a more coarse-grained fashion. However, the available data for water consumption still exhibit extremely low temporal and spatial granularity. Also, they are highly aggregated and complex, limiting the potential for data mining and analysis services.

Given the current low volume of data derived from water measurements, existing systems cannot scale to manage water consumption data. Towards this direction, *smart water meters* are installed either on single fixtures or on a residence supply system and provide us with a wealth of such data. But still, consumers have limited access to analytics and information concerning their usage patterns and habits. Furthermore, personal water monitoring and typical metering infrastructures must be decoupled, but also interoperable. This way, they can provide a better understanding of water use and influence sustainable consumer behaviour.

During the last few years, distributed computing technologies have emerged to satisfy the need for systems that can efficiently manage massive volumes of data. *MapReduce* is a programming model that enables execution of algorithms on a cluster based environment, through mapping elements of a dataset in a key-value pair perception (i.e. *mappers*) to several machines (i.e. *reducers*), via a hashing function. The result is stored in a *Distributed File System* (DFS). Several open-source frameworks of the MapReduce programming model exist, including *Hadoop*[1], *Spark*[2] and *Flink*[3]. Flink can operate over Hadoop DFS (HDFS) and exhibits better performance on MapReduce algorithms compared to the other frameworks [1].

Forecasting tasks are often performed by applying machine learning algorithms over large data collections. The simplicity and the effectiveness of *k-Nearest Neighbours* (k-NN), have fed many research communities with numerous applications and scientific approaches, which evolve its potential over the years. Of particular interest are the k-NN join methods [2], which retrieve the nearest neighbours of *every* element in a testing dataset (R) from a set of labelled elements in a training dataset (S). Each data element consists of several *features*, which constitute the preliminary knowledge on which the classification is conducted. However, computing k-NN joins on huge amounts of data is time consuming when conducted by a single CPU, as it requires k-NN computation for every element in dataset R. The MapReduce model accelerates this computation, allowing for parallel and distributed execution.

[1]https://hadoop.apache.org/
[2]https://spark.apache.org/
[3]https://flink.apache.org/

This work addresses the challenges of Big Data management, analysis and interpretation of massive water consumption data collected by smart meters. It proposes a method for scalable data analysis and knowledge extraction of diverse Big Data collections. It also devises novel means which facilitate consumers and water utilities to draw useful conclusions, lead them towards efficient water usage and induce sustainable lifestyles. We focus on water consumption classification and forecasting, by using large volumes of historical water consumption data.

In this paper, we present the Flink zkNN (*F-zkNN* for short), a robust probabilistic classifier for parallel and distributed execution by extending over the *H-zkNN* [3]. More particularly, we propose a new k-NN based probabilistic classifier which identifies and predicts water consumption class probability for arbitrary temporal basis, by using a voting scheme. We introduce a MapReduce based approach which reduces file operations for large amounts of data and is uniquely initialized upon launch. Our approach is unified in a single session to reduce space occupation and cluster overloading. Through an experimental evaluation we show that the proposed method efficiently achieves high prediction precision and useful knowledge extraction.

II. Related Work

Various approaches have been proposed w.r.t. classification, forecasting and prediction. To achieve these tasks, our work resides in the area of *distributed* and *parallel* computation of k-NN joins over massive amounts of data. In the following, we present the relevant literature review.

The transition to Big Data has led the k-NN problem to be widely discussed in the literature. As the number of dimensions increases, distance computations need an exponentially larger amount of CPU. In our case, the dimensions emerge prohibitively greater and thus the execution of a k-NN joins method on huge data volume requires long-lasting operations. To overcome this issue, we apply *dimensionality reduction* on the data via a *Space Filling Curve* (SFC) based approach, which is significantly faster than other similar methods [4]. In particular, we use the *z-order* curve, whose elements are expressed by the *z-value*. The z-order curve maps a multidimensional set to one dimension and its accuracy lies in the scanning order of the elements. Figure 1a shows the recursive way the z-order curve scans the elements of a two-dimensional space.

Several implementations of MapReduce based k-NN joins algorithms have been proposed in the direction of data mining in a distributed environment. Song et al. [5] review the most efficient approaches among them, concluding that H-zkNN [3], built on a SFC based approach, is significantly faster. The k-NN based knowledge extraction usually includes three stages, i.e. (i) data pre-processing, (ii) data partitioning and organization, and (iii) k-NN computation. Our work extends over the method of Zhang et al. [3] by additionally proposing a robust probabilistic classifier based on the MapReduce programming model, executed in a single parallel session, which achieves high prediction accuracy. We further apply this classifier on forecasting tasks in order to draw useful knowledge from huge amounts of data.

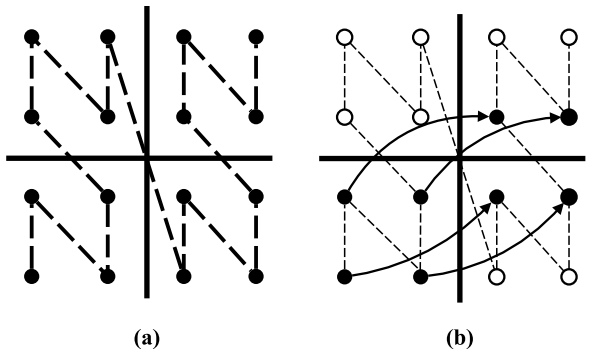

Figure 1: The z-order curve.

Classification and forecasting can be performed by applying a k-NN classifier. It can be realized by predicting the class of a new observation and having already determined the dominant class among its nearest neighbours. Gou et al. [6] present a weighted voting scheme for such classifiers, where the distance between an element and its nearest neighbours determines the weight of each neighbour, with the most weighted one determining the final class of the query element. We extend over this functionality by also computing the probability that an element has, in order to belong to each class.

Similar approaches that apply the k-NN classifier in the resources consumption domain, include the work of Chen et al. [7] who used a k-NN classification method and labelled water data to identify water usage. However, they do not operate on huge amounts of data, while our approach lies in the field of Big Data management. Similarly, Kermany et al. [8], apply the k-NN classifier on low granular water consumption data, whilst this work is applied over highly granular data in a distributed and parallel environment.

III. Preliminaries

In order to efficiently extract valuable knowledge from a huge volume of water consumption data, we apply classification and exploit the MapReduce programming model. In the following, we present some key ideas that our approach brings together.

A. Problem Definition

Classification is applied in order to determine in which class of labelled elements a new observation belongs to. A k-NN classifier is responsible for obtaining the dominant among the k-Nearest Neighbours' classes. To achieve that, we apply a weighted voting scheme and we calculate a probability to each one of the candidate classes.

Let us consider the set of the k-NN as $X = \{x_1^{NN}, x_2^{NN}, ..., x_k^{NN}\}$ and the class of each one as a set $C = \{c_1^{NN}, c_2^{NN}, ..., c_k^{NN}\}$. The weight of each nearest neighbour is calculated as follows:

$$w_i = \begin{cases} \frac{d_k^{NN} - d_i^{NN}}{d_k^{NN} - d_1^{NN}} & : d_k^{NN} \neq d_1^{NN} \\ 1 & : d_k^{NN} = d_1^{NN} \end{cases}, \quad i = 1, ..., k \quad (1)$$

where d_1^{NN} is the distance of the query element to the closest neighbour, d_i^{NN} and d_k^{NN} its distance to the i-th and the furthest nearest neighbour respectively. Thus, the nominator of Equation 1 expresses the distance difference of the furthest neighbour from the i-th neighbour, while similarly, the denominator expresses the distance difference of the furthest neighbour from the closest neighbour to the query element. By this calculation, the closest neighbours will be assigned a greater weight.

Let $P = \{p_j\}_{j=1}^{l}$ be a set containing each probability class, where l is the number of classes. The probability of each class is the sum of the weights of the similarly labelled nearest neighbours, divided by the total weight of the nearest neighbours and is derived as follows:

$$p_j = \frac{\sum_{i=1}^{k} w_i \cdot I(c_j = c_i^{NN})}{\sum_{i=1}^{k} w_i}, \quad j = 1, ..., l \quad (2)$$

where $I(c_j = c_i^{NN})$ is a function which takes the value 1 if the class of the neighbour x_i^{NN} is equal to c_j.

Finally, the element will be classified as the class with the highest probability, according to the following formula:

$$c_r = \arg\max_{c_j} P, \quad j = 1, ..., l \quad (3)$$

where c_r is the resulting class.

B. The H-zkNN Algorithm

The H-zkNN algorithm is a Hadoop based implementation of the k-NN, which operates in three separate sessions and overcomes the complexity issues arising from huge amounts of data. It applies dimensionality reduction on the R and S datasets and efficiently splits the work among several machines.

Similar approaches partition the datasets in n blocks, thus requiring n^2 reducers to calculate each element's nearest neighbours, due to the fact that every possible pair of blocks has to be evaluated. The H-zkNN algorithm overcomes the problem of data partitioning by adopting a sampling approach on R and S datasets. This way, the sampled data can be easily sorted, allowing for partitioning ranges to be determined. Consequently, the reducers can receive subsets of R and S only in a specific range. This task requires only n reducers to perform the calculations instead of n^2 that would be needed in order to evaluate every possible pair of blocks.

Regarding the dimensionality reduction, the H-zkNN algorithm calculates the z-order curve of the input elements, in order to significantly reduce the complexity of the calculations. It does so by interleaving the binary codes of an element's dimensions, which takes place starting from the most significant bit (MSB) towards the least significant (LSB). For example, the z-value of a 3-dimensional element with feature values 3 (011_2), 4 (100_2) and 5 (110_2), can be formed by first interleaving the MSB of each number (0, 1 and 1) going towards the LSB, thus forming a final value of 011101100_2. This procedure does not require any costly CPU execution.

Figure 1a shows how the z-order curve fills a two-dimensional space from the smallest z-value to the largest. It can be noticed that some elements are falsely calculated being closer than others, as the curve scans them first. This in turn creates an impact on the result's precision. The H-zkNN method addresses this by shifting all the elements by randomly generated vectors and repeating the procedure using the shifted values, thus compensating part of the lost precision through scanning the space in an altered sequence. This is demonstrated in Figure 1b. The four bottom-left elements are shifted twice in the x-axis and once in the y-axis, altering the sequence in which they are scanned by the z-order curve. Consequently, taking under consideration nearest neighbours of a point from shifted datasets, one can obtain elements that are close to it, but had been missed by the un-shifted curve. The main drawback of this approach is the fact that it has to be executed multiple times, one for each chosen shift.

The H-zkNN algorithm operates in three separate MapReduce stages each requiring the initiation of a new Hadoop session. Alternatively, the F-zkNN classifier combines the three stages into a single Flink session. In the following section, our approach is presented in more details.

IV. The F-zkNN Probabilistic Classifier

The F-zkNN probabilistic classifier has been unified in a single distributed session over the Flink framework. Flink offers a variety of *transformations* on datasets and is more flexible due to the fact that a task can be carried out by a number of available generic *task managers*, which mostly denote a single machine on a cluster, constituted of several processing slots. The reduce transformation can be executed over grouped datasets, via the *groupBy* operation, which assigns a part of the dataset to a different reducer, according to a pre-specified field. In order to achieve similar functionality to the Hadoop's map and reduce operations, the *FlatMap* and *GroupReduce* transformations are used, which return an arbitrary number of elements, rather than just a single one. Another aspect that makes Flink more appropriate, is that it does not require key-value pairs during the transitions that take place between the transformations. Instead, *Objects* or just primitive types are used, optionally grouped in *Tuples*.

The scheme of the proposed algorithm, depicted in Figure 2, consists of the following three MapReduce stages:

A. The pre-processing stage

During the pre-processing stage, the R and S datasets, along with the random vectors, are read as plain text from

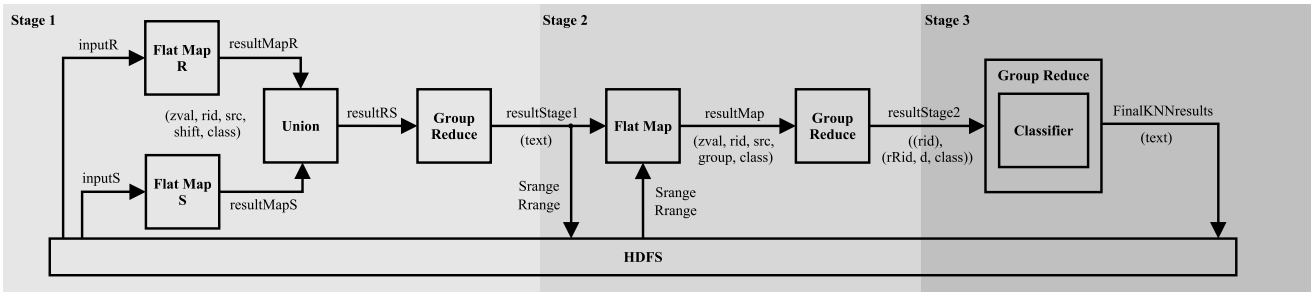

Figure 2: Single session F-zkNN.

the HDFS and delivered to two separate concurrent FlatMap transformations, identifiable by the input source file. For each of the α number of shifts, (Algorithm 1, Line 2), the shifted datasets and their z-values are calculated (Lines 3 and 4) and passed on a *Union* transformation (Line 5). This results in a union of the two datasets, which is passed (as an object containing the values $(zval, rid, src, shift, class)$) on a GroupReduce transformation, grouped by the shift number. During the reduce phase, the datasets are sampled ($\hat{R}_i$ and $\hat{S}_i$) and cached locally (Lines 5-16). The partition ranges ($Rrange_i$ and $Srange_i$) for each shift are calculated using the sampled datasets and are stored on the HDFS (Lines 17 and 18). The output of this stage is the locally cached transformed datasets, which are finally read from the cache and feed the next stage (Line 19).

Algorithm 1: The F-zkNN pre-processing stage.

Input: Datasets R, S and random vectors $\mathbf{V} = \{\mathbf{v}_1, ..., \mathbf{v}_\alpha\}, \mathbf{v}_1 = \overrightarrow{0}$
Output: Transformed datasets R_i^T and $S_i^T, i = 1, ..., \alpha$

1 **begin**
2 **for** $i = 1, ..., \alpha$ **do**
3 $R_i = R + \mathbf{v}_i$, $S_i = S + \mathbf{v}_i$
4 $R_i^T \leftarrow$ CALCZVAL(R_i), $S_i^T \leftarrow$ CALCZVAL(S_i)
5 **foreach** $x \in R_i \cup S_i$ **do**
6 $r \leftarrow$ RANDOM$(0, 1)$
7 **if** $r < MinThreshold$ **then**
8 **if** $x \in R_i$ **then**
9 INSERTSAMPLE$(s, \hat{R}_i)$
10 **end**
11 **else if** $x \in S_i$ **then**
12 INSERTSAMPLE$(s, \hat{S}_i)$
13 **end**
14 **end**
15 STORELOCAL(s)
16 **end**
17 $Rrange_i \leftarrow$ CALCRANGE$(\hat{R}_i)$, $Srange_i \leftarrow$ CALCRANGE$(\hat{S}_i)$
18 STOREHDFS$(Rrange_i, Srange_i)$
19 **return** FETCHLOCAL(R_i^T, S_i^T)
20 **end**
21 **end**

B. The partitioning and pre-calculation stage

In this stage, the transformed datasets are received by the mappers and the previously computed partition ranges are used (Algorithm 2, Line 4) to partition the datasets to $n \times \alpha$ blocks, (Lines 8 and 15), α being the number of shifts and n the number of partitions. Each block is then delivered to a different reducer (Lines 19-28) by using the groupBy operation (as an object containing $(zval, rid, src, group, class)$). There, a range search is performed on each sorted partition and the k-NN of each $x \in R$ element are determined (Line 22). The neighbouring elements' coordinates are then calculated (Line 23), un-shifted using the random vectors (Line 24) and their distance to the $x \in R$ element is computed (Line 25). Finally, they are integrated into the proper dataset (Line 26) grouped by element, along with the calculated distance and feed the final stage (Line 30).

Algorithm 2: The F-zkNN partitioning and pre-calculation stage.

Input: Datasets R_i^T, S_i^T, $i = 1, ..., \alpha$
Output: Dataset $R_{kNearest \times \alpha}$

1 **begin**
2 $R_{kNearest \times \alpha} = \emptyset$
3 **for** $i = 1, ..., \alpha$ **do**
4 READHDFS$(Rrange_i, Srange_i)$
5 **foreach** $x \in R_i^T$ **do**
6 **for** $g = 1, ..., n$ **do**
7 **if** ZVAL$(s) \in Rrange_i(g)$ **then**
8 ADDINTOPARTITION$(s, R^{g \times i})$
9 **end**
10 **end**
11 **end**
12 **foreach** $x \in S_i^T$ **do**
13 **for** $g = 1, ..., n$ **do**
14 **if** ZVAL$(s) \in Srange_i(g)$ **then**
15 ADDINTOPARTITION$(s, S^{g \times i})$
16 **end**
17 **end**
18 **end**
19 **for** $g = 1, ..., n$ **do**
20 SORT$(R^{g \times i})$, SORT$(S^{g \times i})$
21 **foreach** $x \in R^{g \times i}$ **do**
22 $RES \leftarrow$ RANGESEARCH$(s, kNearest, S^{g \times i})$
23 $CC \leftarrow$ CALCCOORDS(RES)
24 $US \leftarrow$ UNSHIFT(CC)
25 $CD \leftarrow$ CALCDIST(s, US)
26 $R_{kNearest \times \alpha} \leftarrow$ ADD$(s \times CD, R_{kNearest \times \alpha})$
27 **end**
28 **end**
29 **end**
30 **return** $R_{kNearest \times \alpha}$
31 **end**

C. The k-NN stage

In the final stage, we directly propagate the second stage's results to the reducers. This increases the calculation efficiency as it reduces the resource requirements of the execution. Thus, the calculated $\alpha \times k$-NN of each R element, are received in this stage's reducers (Algorithm 3, Lines 3-7), which perform $|R|$ reduce tasks via a Tuple containing a string value and an object containing $(rid, rRid, d, class)$. The k-NN of each R element are fetched from the grouped set of $R_{k \times \alpha}$. After determining its final nearest neighbours (Line 4), each query element is classified (Line 5) according

to the probability of each class, which is calculated by using the Equation 3. Finally, the results are added to the resulting dataset (Line 6), which is then stored on the HDFS (Line 8).

Algorithm 3: The F-zkNN k-NN stage.

Input: Datasets $R, R_{kNearest \times \alpha}$
Output: Stored dataset R_f on the HDFS

```
1 begin
2     R_f = ∅
3     foreach x ∈ R do
4         RES ← k-NN(x, R_{kNearest×i})
5         C ← CLASSIFY(RES)
6         R_f ← ADD(s, C, R_f)
7     end
8     STOREHDFS(R_f)
9 end
```

V. Experimental Evaluation

The F-zkNN probabilistic classifier was evaluated on future water consumption forecasting and the extraction of useful knowledge from consumption time-series data, collected in a city scale by smart water meters in Switzerland and Spain.

A. Experimental Setup

The experimental evaluation was conducted on a distributed and parallel environment. The setup includes a system with 4 CPUs each containing 8 cores clocked at 2.13GHz, 256GB RAM and a total storage space of 900GB. The total parallel capability of the system reaches the 64 threads. The F-zkNN algorithm is executed by using 16 task managers with 8192MB heap size each.

We firstly experimented on a real-world dataset coming from Switzerland (Switzerland dataset for short). It included shower events from 77 households, each containing an identifier, a timestamp, a smart water meter measurement calculated in litres, the average temperature and demographic information. This information was related to the age, income, number of males or females and total number of household members. This dataset counts 5795 records. To increase its volume and assess the classifier's prediction efficiency, we used the *BigDataBench*[4], which is a big data generator. We created various synthetic dataset sizes, scaling from 50K records to 15M records.

In the next step, we experimented on large-scale real-world smart water meter data coming from Spain (Spain dataset for short). The water consumption data were formed in hourly time-series coming from 1000 households and covering a time interval of a whole year, i.e. from July 2013 to June 2014. The records included an identifier, a timestamp and a smart water meter measurement calculated in litres. This dataset was constituted by 8.7M records.

The optimal value of the k parameter for the F-zkNN probabilistic classifier was determined through an experimental investigation. The best choice in our context and datasets was proven to be the value of 15. To overcome the lost precision due to the sampling stage, two shifts were adequate on the datasets.

B. Feature Selection

Water consumption time-series pose several challenges on applying machine learning algorithms for classification and forecasting. Thus, proper features that represent and correlate different aspects of the data need to be defined in order to also draw useful conclusions about the determinants that affect water consumption.

Regarding the Switzerland dataset, which included demographic information, we used a dataset of increased volume generated by the BigDataBench (15M records). We took into consideration the sex, the age and the income of the person that generated the shower event. We assessed the classifier by using binary classification for these three features. The sex prediction was made by using the showers for which we knew whether the person was male or female, i.e. households with only one, or of the same sex inhabitants. The age prediction involved the determination of whether the person that generates the shower event is of age less than 35 years, or not. Finally, the income prediction involved the identification of whether the person that takes the shower has income less than 3000 CHF or not.

Concerning the Spain dataset (8.7M records), we took into consideration several temporal and seasonal features, expressed by different time intervals, which affect the water usage and demand, i.e. the daily time-zone (i.e. [09:00-17:00), [17:00-21:00), etc.), the hour, the weekday, the month and the season. According to each household's mean monthly water consumption in litres, the customers' water usage was categorized from "very environmental friendly", to "significantly spendthrift". By averaging the hourly water consumption and determining the highest and lowest value, five equally ranged, demand classes were determined, i.e. "minimum" ($<$6 litres), "low" ($\geq$6 and $<$15 litres), "normal" ($\geq$15 and $<$23 litres), "high" ($\geq$23 and $<$40 litres) and "maximum" ($\geq$40 litres). Besides, we integrated weather conditions via the *Weather Underground* API[5]. The integrated weather conditions included the hourly temperature, humidity, and continuous rainfall or heat.

C. Quantitative Evaluation

The F-zkNN probabilistic classifier was evaluated in the direction of prediction accuracy and useful knowledge extraction, for both datasets. The algorithm was evaluated by forecasting water consumption classes for specific time intervals and also by drawing useful conclusions related with user characteristics (i.e. sex, age and income). Due to space shortage, the figures show results either for forecasting or for knowledge extraction.

The classifier regarding the forecasting was evaluated over the Spain dataset and the prediction of next day's hourly consumption of one or more households for a given time

[4]http://prof.ict.ac.cn/BigDataBench/

[5]http://www.wunderground.com/

interval. Figure 3 shows a visualization of the classifier's output. The hourly water consumption characteristics are distributed into classes. The bars indicate the hourly probability of one or more households water consumption to be characterised by each demand class.

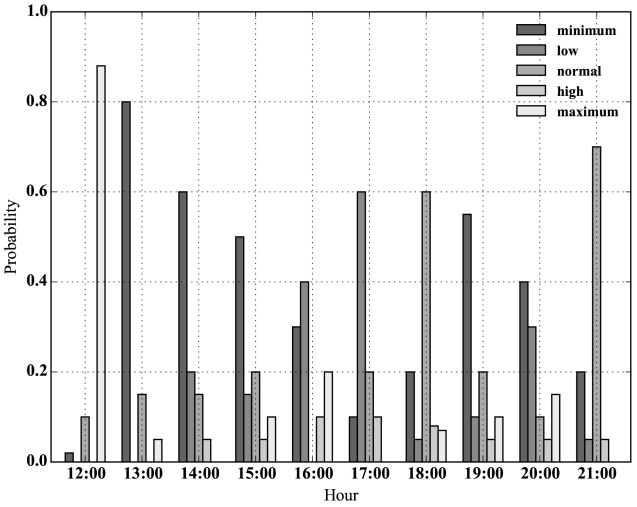

Figure 3: Hourly forecasting of each consumption class.

Regarding the Switzerland dataset, we used the ten-fold *cross-validation* approach in order to assess useful knowledge extraction. To achieve that, we iteratively split each dataset into ten equal parts and executed the algorithm the same number of times, using a different subset as training set (R) and the rest of the sets, unified, as testing set (S). The classifier achieved a prediction precision of 78.6%, 64.7% and 61.5% for sex, age and income, as illustrated in Figure 4. The results provided us with useful insights concerning water consumption determinants. An important determinant of shower water demand is hair length. Females who have longer hair consume more water. The wrong sex predictions (i.e. 21.4% out of 100.0%), is due to the variable hair length of each sex. Moreover, age and income are less important determinants than sex, as people consume water for their daily needs, regardless their age and income. People with a decent income can afford having more expenses regarding their water consumption. Similarly, younger people are well informed about the environmental sustainability, resulting in less wasteful showers.

VI. Conclusions

This work describes a novel approach delivering a MapReduce based k-NN probabilistic classifier. Firstly, we applied data dimensionality reduction and ensured efficient data sorting and distance computations. We proceeded by performing data partitioning and pre-calculation, which fed the k-NN based data classification process. Lastly, we conducted an experimental evaluation to assess the algorithm w.r.t forecasting, prediction precision and useful knowledge extraction.

The directions for future work include the performance comparison of the the F-zkNN algorithm with other similar approaches to draw useful conclusions and quantify its efficiency in terms of scalability and execution time. Further experimental investigations will focus on the improvement of classifier's precision and the dynamic determination of the consumption classes by exploiting the data characteristics.

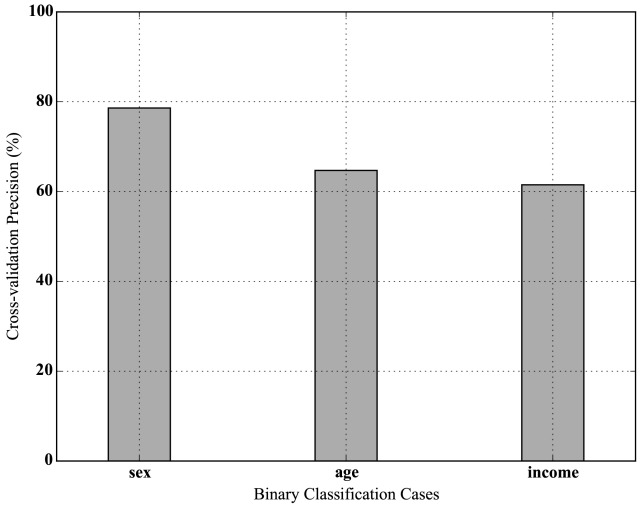

Figure 4: Prediction precision for sex, age and income.

Acknowledgment

This work was supported by the EU FP7 Collaborative Project - DAIAD (FP7-ICT-2013-619186).

References

[1] J. Galilee and Y. Zhou, "A study on implementing iterative algorithms using big data frameworks," in *School of IT Research Conversazione Posters*, 2014.

[2] C. Böhm and F. Krebs, "The k-nearest neighbour join: Turbo charging the kdd process," *Knowledge and Information Systems*, vol. 6, no. 6, pp. 728–749, 2004.

[3] C. Zhang, F. Li, and J. Jestes, "Efficient parallel knn joins for large data in mapreduce," in *Proc. of EDBT*, 2012, pp. 38–49.

[4] S. Liao, M. Lopez, and S. T. Leutenegger, "High dimensional similarity search with space filling curves," in *Proc. of IEEE ICDE*, 2001, pp. 615–622.

[5] G. Song, J. Rochas, F. Huet, and F. Magoulès, "Solutions for processing k nearest neighbor joins for massive data on mapreduce," in *Proc. of PDP*, 2015.

[6] J. Gou, T. Xiong, and Y. Kuang, "A novel weighted voting for k-nearest neighbor rule," *JCP*, vol. 6, no. 5, pp. 833–840, 2011.

[7] F. Chen, J. Dai, B. Wang, S. Sahu, M. Naphade, and C. Lu, "Activity analysis based on low sample rate smart meters," in *Proc. of ACM SIGKDD*, 2011, pp. 240–248.

[8] E. Kermany, H. Mazzawi, D. Baras, Y. Naveh, and H. Michaelis, "Analysis of advanced meter infrastructure data of water consumption in apartment buildings," in *Proc. of ACM SIGKDD*, 2013, pp. 1159–1167.