Elena Ivannikova

SEMANTIC PLACE RECOGNITION FOR CONTEXT AWARE SERVICES

JYVÄSKYLÄN YLIOPISTO
TIETOTEKNIIKAN LAITOS
2012

TIIVISTELMÄ

Ivannikova, Elena
Paikkojen semanttisen merkityksen ennustaminen matkapuhelimen dataan perustuvia
Jyväskylä: Jyväskylän yliopisto, 2012, 66 s.
Tietotekniikka, pro gradu -tutkielma
Ohjaaja(t): Hämäläinen, Timo

Merkityksen selvittäminen tärkeimmistä paikoista, joissa käyttäjät usein vierailevat on relevantti ongelma mobiilissa tietojenkäsittelyssä. Kyseisten paikkojen semanttisen merkityksen ennustamista tarvitaan useilla eri aloilla. Käyttäjän sijainnin semanttinen merkintä on myös haastavaa palveluntarjoajille. Tietoisuus käyttäjien toimista on tärkeää personoitujen palvelujen kehitykselle muun muassa terveydenhuollonjärjestelmissä sekä ihmisten päivittäisessä elämässä. Mobiilikäyttäjien sijainnin ennustaminen mahdollistaa sekä korkeatasoisten paikkatietopohjaisten palveluiden ja sovellusten kehittämisen että tehostaa resurssointia langattomissa verkoissa. Tässä tutkimuksessa esitellään ja analysoidaan useita matkapuhelimen dataan perustuvia semanttisen sijainnin ennustuksen ratkaisuja. Esitellyt menetelmät ovat testattu yli sadan käyttäjän pitkällä aikavälillä keräämällä datalla.

Avainsanat: semanttinen paikan ennustus, semanttinen sijainti, paikkatietoon pohjautuvat palvelut, tiedonlouhinta, luokittelu, mobiili tietojenkäsittely

ABSTRACT

Ivannikova, Elena
Semantic place prediction from the mobile phone data
Jyväskylä: University of Jyväskylä, 2012, 66 p.
Information Technology, Master's Thesis
Supervisor(s): Hämäläinen, Timo

Extracting the meaning of the most significant places, which are frequently visited by a mobile user, is a relevant problem in mobile computing. Predicting semantic meaning of such places is useful in many areas. The problem of place semantic annotation of a user location can be challenging for service providers. Awareness of user activities is very important for development of personalized applications, which can be used in health care systems, living systems, etc. Predicting location of mobile users not only enables development of high quality location-based services and applications, but also improves resource reservation in wireless networks. In this research several solutions for semantic place prediction from mobile phone data are suggested and analyzed. Presented approaches have been tested on a real dataset collected during long period of time involving more than a hundred of participants.

Keywords: semantic place prediction, semantic location, location-based services, data mining, classification, mobile computing

GLOSSARY

CAA	Context Aware Applications
CAS	Context Aware Services
FN	False negative
FP	False positive
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HMM	Hidden Markov Model
LBA	Location Base Applications
LBS	Location Based Services
MMC	Mobility Markov Chain
MMM	Mixed Markov-chain Model
PCA	Principal Component Analysis
RAM	Random Access Memory
SOM	Self-Organizing Maps
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
WiFi	Wireless Fidelity
WLAN	Wireless Local Area Network

LIST OF FIGURES

FIGURE 1 Methodological approach 12
FIGURE 2 A simple Kohonen network 24
FIGURE 3 Graphical model of a Naïve Bayes classifier 27
FIGURE 4 Data collection 34
FIGURE 5 Principal components of the data set D2, ordered by their variance. 42
FIGURE 6 Correlation values of D1 features 44
FIGURE 7 Results of the smoothing parameter validation for the Naïve Bayes (a) original size, (b) enlarged size 45
FIGURE 8 Results of the Naïve Bayes classification for the dataset D1_reduced 45
FIGURE 9 Number of correctly and incorrectly Naïve Bayes classified data for the data set D1_reduced, grouped by labels. 46
FIGURE 10 Percentage of correctly and incorrectly Naïve Bayes classified data for the data set D1_reduced, grouped by labels. 46
FIGURE 11 Results of grid search for the SVM 48
FIGURE 12 Results of the SVM classification for the dataset D2_reduced 49
FIGURE 13 Number of correctly and incorrectly SVM classified data for the data set D2_reduced, groupped by labels 50
FIGURE 14 Percentage of correctly and incorrectly SVM classified data for the data set D2_reduced, groupped by labels 50
FIGURE 15 Results of the Map Grid Size parameter validation for the SOM 51
FIGURE 16 Results of the Supervised SOM classification for the dataset D2_reduced 52
FIGURE 17 Number of correctly and incorrectly Supervised SOM classified data for the data set D2_reduced, groupped by labels 53
FIGURE 18 Percentage of correctly and incorrectly Supervised SOM classified data for the data set D2_reduced, groupped by labels 53
FIGURE 19 Comparison of the evaluation file and Naïve Bayes plus instance-wise "voting" classification 55
FIGURE 20 Comparison of the evaluation file and SVM plus instance-wise "voting" classification 55
FIGURE 21 Comparison of the evaluation file and Supervised SOM plus instance-wise "voting" classification 56

LIST OF TABLES

TABLE 1 Place labels 35
TABLE 2 Place labels ordered by presence in the data set (in %) 41
TABLE 3 Performance evaluation results of (a) Naïve Bayes plus instance-wise "voting", (b) Supervised SOM plus instance-wise "voting", (c) SVM plus instance-wise "voting" 57

TABLE OF CONTENTS

TIIVISTELMÄ 2

ABSTRACT 3

GLOSSARY 4

LIST OF FIGURES 5

LIST OF TABLES 6

TABLE OF CONTENTS 7

1 INTRODUCTION 9
1.1 Description of problem area 9
1.2 Research problem statement 11
1.3 Methodology 11
1.4 Related work 12
1.5 Thesis structure 15

2 DATA PREPROCESSING 17
2.1 Feature selection 17
2.2 Normalization 18
2.3 Dimensionality reduction 19
2.3.1 Principal Component Analysis description 20
2.3.2 Principal Component Analysis algorithm 21
2.3.3 Principal Component Analysis evaluation 21

3 METHODS DESCRIPTION 23
3.1 Supervised Self- Organizing Maps 23
3.1.1 SOM method description 23
3.1.2 SOM algorithm description 24
3.2 Naïve Bayes 25
3.3 Support Vector Machine 28
3.3.1 SVM method description 28
3.3.2 SVM tools description 29
3.4 Validation and Evaluation 31

4 HANDLING DATA 34
4.1 Data description 34
4.2 Data postprocessing 36
4.2.1 Evaluation 36

4.2.2 Integration 37

5 EMPIRICAL PART 38
5.1 Planning of the machine learning pipeline 38
5.2 Application of the methods 38
5.2.1 Feature selection 39
5.2.2 Normalization 41
5.2.3 Principal Component Analysis 42
5.2.4 Naïve Bayes 43
5.2.5 Support Vector Machine 47
5.2.6 Supervised SOM 51
5.3 Analysis of the results 54

6 CONCLUSIONS AND FUTURE WORK 59

REFERENCES 61

1 INTRODUCTION

1.1 Description of problem area

With the increasing pervasiveness of smartphones over the past years, Location-Based Services (LBS) and Location-Based Applications (LBA) received an increasingly close attention. The LBSs and LBAs operate based on knowing user's context, and thus, also called Context-Aware Services (CAS) and Context-Aware Applications (CAA). The concept of context in this case assumes two things: (1) a collection (possibly historical) of measurements or logs retrieved from a mobile device (time, accelerometer, landmarks in the vicinity of geographical location of the user, e.g., GPS, GSM, Wi-Fi beacons, and mobile device profile, e.g., calendar entries, messages, calls, media, applications) and (2) semantic or behavioral meaning of a location or semantic location. While the first aspect of the context refers to the features of the physical context that are directly measurable (or trivially inferred, e.g., objects that are in the vicinity of the user), the second aspect is related to what the user does in this location or what is the meaning of the location for the user.

Knowledge of user activities is very important for development of personalized applications, which can be used in health care systems, living systems, etc. It is crucial for mobile content providers, who can use this information for better interaction with users according to their preferences and current context. Predicting location of mobile users not only enables development of high quality location-based services and applications, but also improves resource reservation in wireless networks. In particular, active research is devoted to such application areas as social-networking, business needs, entertainment, advertisement, healthcare, local traffic, local restaurants, everyday assistants, reminders, meeting scheduling, multi-user collaboration etc. (Kortuem et al., 1999; Marmasse and Schmandt, 2000; Roth and Unger, 2001; Hudson et al., 2002; Terry et al., 2002; Hull et al., 2006; Yoon et al., 2007; Hoh et al., 2008; Mohan et al., 2008; Kim et al., 2011).

As the user's context is not provided to CAA automatically, in order to act relevantly CAA must infer the user's context. There are two scenarios of the inference and subsequent use of the context by an application. The application might want to know (1) the current context or (2) to predict the next context. The first situation is peculiar to reflexive applications that act on a current context. These applications may certainly lack the power of predicting the next visiting location. The second scenario is symptomatic to proactive applications, like everyday assistants or advertisements (Kim et al., 2011). In the case of advertisements this could be important to provide more relevant targeting by considering not only the spatial, but also temporal relevance of advertisements to mobile users.

To identify the current physical location sensing capabilities of mobile devices can be employed. Thus, in-door and out-door localization can be sensed using GPS, Wi-Fi, or simply mobile cell tower / base station identification. Importantly, to increase energy-efficiency the periodicity of scanning the environment using Wi-Fi or GPS should be adapted to how fast a user actually moves. For example, the moving state or speed and character of the move can be identified using the measurements of the multi-axis accelerometers.

In the present study measurements of GPS are not used. This is motivated partially by energy-efficiency considerations. Active use of GPS causes fast battery drain of a mobile device (Zhuang et al., 2010). Additionally use of GPS for context-aware applications raises privacy concerns. Temporal and velocity information (e.g., stopping time, duration of transition, time of an event, speed of transition) provide additional power to location segmentation from the stream of measurements compared to when only spatial information is used in isolation. Moreover temporal and velocity clues may allow inferring the semantic meaning of locations, and efficient modeling of user behavior.

It is important to differentiate between places and locations concepts. While location term is related to a physical location, the place is a more abstract term unifying several close locations by the similar semantic meaning. Thus, first step in the location prediction algorithms is to extract meaningful locations. Usually this is done by clustering locations into places. Semantic location is defined as a location that has a special meaning for a user, e.g., home, work, school etc. Most of the research works on context-aware platforms are centered around figuring out important locations, routes, and user mobility patterns without explicit assignment of class labels (e.g., home, work, leisure activities etc.). Little attention was paid to inferring semantic meaning of a location, based on spatio-temporal mobility patterns. Moreover, typical data that is used for inferring the context is restricted to only spatial, temporal, and velocity (e.g., current time, GPS, Wi-Fi and cellular signals, accelerometer measurements) features available through a mobile device. Relatively little work has been done on incorporating the mobile device profile information (calendar entries, calls, applications) to existing platforms for mining semantics of a location, i.e., assigning semantic class labels to locations.

This research is focused on predictive and classification capabilities of the information obtained from a mobile device profile without explicitly modeling the user's transition behavior, mobility pattern or semantic behavior.

1.2 Research problem statement

The main aim of this Master's thesis is to develop a solution for predicting semantic meaning of a set of places extracted from mobile phone data and compare different classification techniques.

1.3 Methodology

This section is devoted to methodology used in the current research. All stages the research process has passed are listed and described, and the connections between the research phases are presented.

The main stages through which the research process has passed are problem relevance, solution design and evaluation.

Problem relevance is the first stage, which starts from the problem identification and motivation for the solution development. Initial research of available literature on mobile data analysis and place prediction has been performed. The areas poorly investigated have been identified and explored.

Solution design is the second stage of the research. It is devoted to the solution development for the research problem. This stage includes selection of tools and analysis of the data. A literature review related to data mining, namely normalization, dimensionality reduction, classification, relevant data mining methods and their mathematical backgrounds, mobile data management and fault detection techniques has been performed. Relevant existing tools have been identified, analyzed and compared for the purpose of further use. Required non-existent tools and method have been implemented.

Evaluation is empirical part of the research. At this stage experiments, including feature extraction, classification, validation, and performance estimation have been performed. In order to increase classification accuracy, different data sets have been created, and different classification method have been used.

The final part of the research covers analysis and comparison of the achieved results. Conclusions have been made based on the information obtained at the previous stages. The best solution has been identified.

For better perception Figure 1 provides structural representation of the methodological approach. The sections in blue correspond to the research stages described above. Grey blocks under each section name contain brief description of activities performed at every stage. Arrows represent inter-connection between stages of the research.

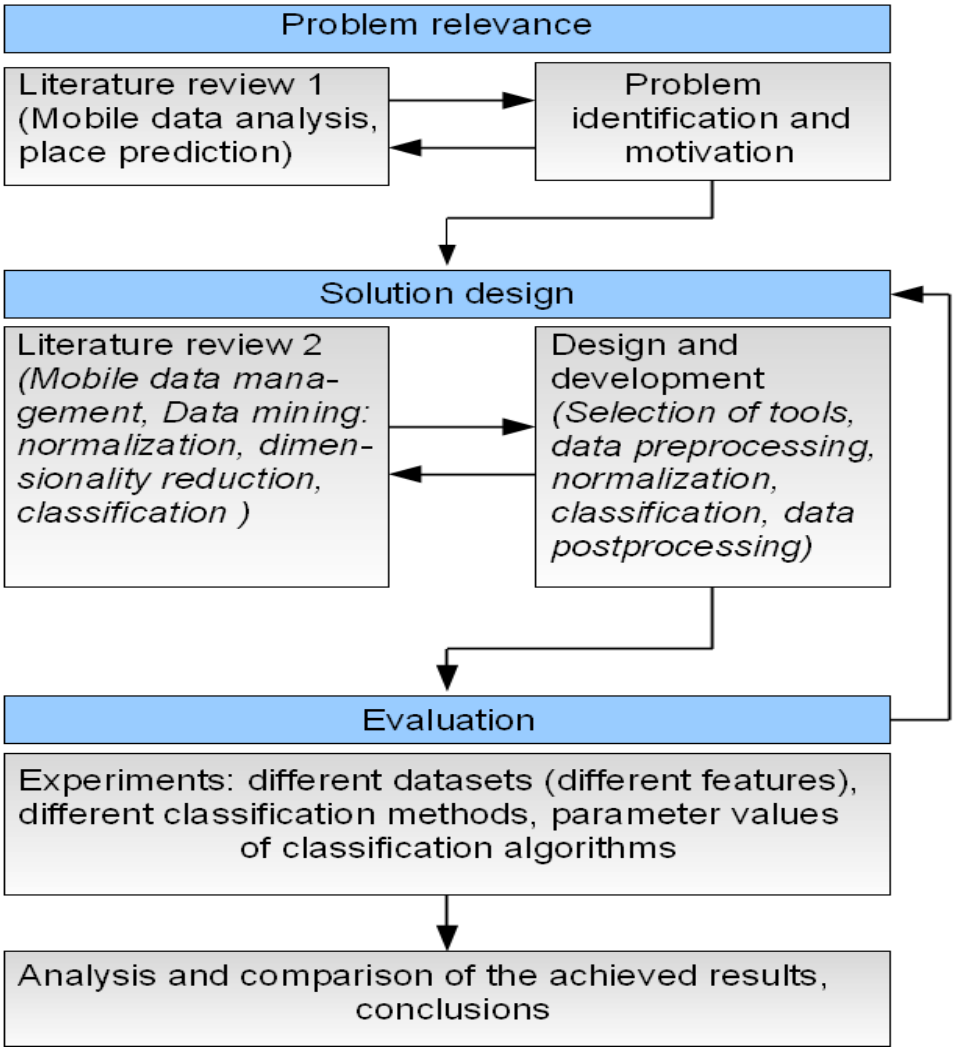


FIGURE 1 Methodological approach

1.4 Related work

This section provides an overview of existing research in the area of analysis of mobile phone data and location prediction. It examines available research topics, models and techniques used and discusses what has been and remains undone so far.

To predict next location, probabilistic graphical models and probabilistic inference are typically employed (Ashbrook and Starner, 2003; Gambs et al., 2010, 2012; Asahara et al., 2011). Little attention has been to extracting the location semantics using the same mechanisms (Kim et al., 2011). To infer the higher-level context or semantics of the location, discriminative models and classification techniques (Duda et al., 2000; Hastie et al., 2009), like SVM (Cristianini and Shawe-Taylor, 2000; Steinwart and Christmann, 2008), most likely to be used in addition to the mobility pattern discovery methods.

Marmasse and Schmandt (2000) use the loss of GPS signals to detect buildings. Lost and later re-acquired within a certain radius GPS signal is considered to be indicative of a building. Wolf et al. (2001) investigate automatic travel diaries. They use stopping time to mark the starting and ending points of trips. Sparacino (2002) proposed a museum guiding system that infers user's behavior type (e.g., greedy, selective, busy) based on infrared beacons and suggests routes. The behavior is modeled using Bayesian networks. The inputs to the model are visitor's stopping times at each exhibit. Bhattacharya and Das (1999) address the issue of limiting the cellular infrastructure resources dedicated to locating a user sufficient for call delivery. Their cellular user-tracking system for call delivery uses transitions between mobile cells as inputs to Markov model. Depending on the user's movement mode (passing between cells or staying in one cell), system tries to assign fewer resources to successfully deliver a call. Similar issues are addressed in Liu and Maguire (1995), where resource management system uses predictive model based on updates of locations provided by mobile devices. System allows services and data to be pre-cached at the most likely future locations.

In Davis et al. (2001) a communication between static and dynamic nodes in ad-hoc networks is managed by a location modeling and prediction system that provides smart and timely resource allocation to ensure reliable and fast packet delivery.

Ashbrook and Starner (2003) present a system that clusters GPS measurements into meaningful locations at multiple scales. Clustering is done using k-means algorithm and extracts locations and sub-locations. The radius of k-means algorithm is identified by finding a knee in the graph of the number of clusters found dependent on the cluster radius. The locations are then used in a Markov model to predict the next user's location or move. They do not take time into account in their model. Their system does not automatically label semantically inferred locations and just learns significant or meaningful places. The system is user-tailored and does not support real-time learning.

An important part of location modeling is the identification of a user's motion mode, e.g., static, driving, walking, bicycling etc. This problem is tackled in Kantola et al. (2010), where authors employ a recursive probabilistic inference over a Hidden Markov Model (HMM). The model obtains evidence from GPS and accelerometer data.

Similar problem is addressed in Patterson et al. (2003), where authors use Bayesian network models for guessing transportation modes (bus, car, foot) based on GPS traces. The input variables they use in their classification model are average speed and variance. They demonstrate that prediction accuracy can be improved by taking into account prior knowledge, e.g., location of bus stops and bus routes.

Liao et al. (2005) proposed a relational Markov network for predicting user's activity given location evidence in a form of GPS data. The authors advocate the idea that user's activities are recognizable from location and time information.

Gambs et al. (2010) propose a Mobility Markov Chain (MMC) algorithm for next location prediction. Essentially they use the same probabilistic model as authors in Ashbrook and Starner (2003), however, they replace the k-means clustering algorithm with the DJ-cluster algorithm (Zhou et al., 2004) specifically tailored for geo-location data.

Kim et al. (2011) propose a visit-pattern-aware mobile advertisement system enabling mobile advertising based on user behavior in commercial complexes. They develop a probabilistic prediction model that predicts user's next visit place from their place visit historical records. Prediction model is based on a Bayesian network that predicts the next visit place by learning sequential visit patterns. Importantly visit patterns are learned in a collective manner, meaning that the model learns common sequential visit patterns from many people and thus, generalizes for many users, including unseen ones. The model includes the following variables: visit place, visit time, visit duration, gender and age. Visit place variable represents possible places explicitly (e.g., cafe, cinema, shop etc.). The model includes variables related to the current period of time and several previous states. Although this model does not design various mobility patterns explicitly, it allows capturing different types of user behavior intrinsically. The system well addresses the issue of both spatial and temporal ads relevance.

Gambs et al. (2012) explore the problem of the next location prediction. They extend a Mobility Markov Chain (Gambs et al., 2010) mobility model to n-MMC by including the n previous location visited to the original model.

A Mixed Markov-chain Model (MMM) for next place prediction was developed in Asahara et al. (2011). MMM is essentially represented by a set of conventional Markov chain models each fit to predict transitions for user group with a common specific mobility pattern. Thus, the first step in this approach is to identify the group a particular user belongs to. Having fixed the group, next place is predicted using this group's Markov chain model.

On the contrary to many batch off-line algorithms for learning semantic locations that have limited practical application Zhang et al. (2007) proposed an online adaptive technique for learning significant locations from GPS data without assigning a semantic class label. Their approach models user candidate locations as a mixture of Gaussians that are updated by a recursive algorithm, once new measurements are taken. The approach is user-tailored as a separate model is learned for each user. In addition to learning of meaningful locations authors proposed an efficient mini-max criterion to discover frequent routes of a user. These procedures allowed authors to develop an automatic traffic advisory system for cell phones.

Kang et al. (2005) propose another algorithm for extracting significant places from a trace of coordinates. Location coordinates are taken by listening for RF-emissions from known radio beacons in the environment (e.g., Wi-Fi access points, GSM cell towers). Authors propose simple yet efficient joint time-location-based algorithm for clustering coordinates into locations. It exploits the

fact that both location visiting duration and visiting frequency are representative for importance quantification.

Bayir et al. (2010) share a substantial amount of work on developing a framework for discovering mobility profiles of cell phone users. Importantly they consider spatial and temporal information jointly to discover spatio-temporal patterns. Spatial information processed is in the form of cell-based location data logs. Their framework consists of five phases: path construction, cell clustering using a weight based hierarchical graph clustering algorithm, topology construction, pattern discovery, post processing. For pattern discovery authors use a Sequential Apriori Algorithm that is a modification of AprioriAll (Agrawal and Srikant, 1995) string analysis based technique.

In Papliatseyeu and Mayora (2008) a hybrid approach for recognition and prediction of mobile user places and activities is suggested. The idea of this approach is to combine data obtained from multiple positioning techniques available in smartphones. Following this approach places are recognized based on raw fingerprints obtained from GSM and Wi-Fi. Also available GPS coordinates can be assigned. Such an approach based on the utilization of a fusion of positioning techniques increases coverage and accuracy. For improvement of places classification it is suggested to use an "importance" rank based on the common-sense reasoning.

Anagnostopoulos et al. (2011) argue that the most suitable algorithms for location prediction are offline and online clustering and classification. Offline kMeans is a suggested algorithm for offline clustering, while Online kMeans and Adaptive Resonance Theory are used to realize online clustering. Classification is done via a Hausdorff-like distance.

According to Isaacman et al. (2011) people spend most of their time around a few key places. Ability to identify important places and movement patterns can be of a significant importance for such areas as deployment of transportation infrastructure and telecommunications. Based on the anonymized Call Detail Records data from a cellular network collected from a number of users, the authors propose a set of methods derived from a logistic regression-based analysis in order to identify important places and apply semantic meaning for the locations.

To conclude there is significant amount of research papers related to analysis of the mobile phone data. However, the majority of works are based on analysis of data obtained through positioning techniques. Only a few papers describe attempts to analyze mobile phone data without location measurements.

1.5 Thesis structure

This Master's thesis contains seven chapters. The current chapter is an introduction and is devoted to overview of the problem area and related research work in the scientific field. Chapter 2 provides description of the data preprocessing techniques. Essential steps of the data preprocessing phase of data anal-

ysis, namely feature selection, normalization, and dimensionality reduction are discussed. Chapter 3 describes classification methods, used in the current research, as well as existing validation and evaluation techniques to be used along with the methods. In Chapter 4 a detailed data description is provided, also a list of challenges related to the data is presented. This chapter also covers data postprocessing phase of data analysis. Then, empirical part of the research work is expressed in the Chapter 5, which describes machine learning pipeline and the approaches to the data classification. Practical results of the suggested solutions implementations are presented in this chapter. Final results are compared, evaluated, and analyzed. Finally, Chapter 6 concludes the whole work and provides ideas for future work.

2 DATA PREPROCESSING

Data preprocessing is initial and important step in any data analysis. The idea of data preprocessing is to prepare input for further data analysis using data mining and machine learning techniques. It is first necessary to integrate data from different sources, bring all the data together into a set of instances and specify attributes. Secondly, it is required to standardize the data set obtained, handle missing and inaccurate values, select a subset of the attributes for learning the model and minimize noise in the data set. This chapter describes data preprocessing steps performed in the current research.

2.1 Feature selection

In machine learning data is the result of measurements, which are often represented by a combination of items placed in rows and combination of multiple variables placed in columns.

Many factors affect efficiency of machine learning algorithms for a given task. The quality of data is one of such factors. Feature selection, or attribute selection, is a process of selecting a subset of the most important features to build a robust and accurate learning model. It is very important particularly in those areas, where a large number of measured variables combine with relatively small number of available samples (or instances). It is also useful if a data set contains missing values or skewed inputs.

The initial step of feature selection is manual analysis following by more advanced techniques for minimization of noise. In the stage of manual analysis, it is important to have comprehensive knowledge of problem area and clear understanding of the given task. Based on these skills and personal understanding of the data set and meaning of the measurements, a researcher can manually select a subset of needed features. In this phase the features can be measurements themselves, combinations of measurements or other measurements transformations.

The next step relates to handling missing and inaccurate values, and processing values of different types, such as continuous, discrete and textual data. It is important to distinguish between data types, because they influence techniques, which can be used for data analysis.

If needed, following the stage of manual feature selection, other more advanced dimensionality reduction techniques may be applied.

Feature selection assists with improving performance of learning models through a careful process of a data set measurements transformations and removing most irrelevant and redundant features.

2.2 Normalization

Normalization is an important part of data pre-processing. Normalizing data helps to make the source data internally consistent. Due to normalization each data type has the same kind of content and format. Normalization makes data dimensionless. It is necessary to apply normalization when the dissimilarity measure is sensitive to the differences in the magnitudes or scales and when one variable can mask others.

Formal definition of normalization can be described as follows:

$$x_{ij}^{*} = \frac{x_{ij} - L_j}{M_j},$$

where $\mathrm{X} = \{x_{ij}\}$ represents raw data, L_j reflects location measure and M_j relates to scale measure. Depending on choosing different location and scale measures, various normalization methods can be obtained. There are different standardization techniques, but only several methods will be described here.

Z-score normalization is used for transformation of normal data to standard score form. This method is useful when it is needed to determine whether a specific value is above or below average, and by how much. In z-score normalization each value of the obtained numeric matrix is calculated from values of the current matrix as follows:

$$x_{ij}^{*} = \frac{x_{ij} - \mu_j}{\sigma_j}.$$

Here μ_j is the mean value of the j-th column and σ_j is the j-th variable standard deviation. Each column of the new matrix has a mean of zero and a variance of one. Z-score is useful when comparing variables with very different observed units of measure.

Range (min-max) normalization transforms the numbers in a column to fit within a specific range. Often a data set is transformed to numbers between zero and one. This type of normalization is helpful for ensuring that extreme values are located within a fixed range. In this method the range of the variable

is used for normalization and the elements of a new matrix can be calculated in the following way:

$$x_{ij}^{*} = \frac{x_{ij} - \min_{j} x_{ij}}{\max_{j} x_{ij} - \min_{j} x_{ij}}.$$

In the obtained matrix the lowest value is set to 0 and the highest value is set to 1. This provides an easy way to compare values that are measured using different scales or different units of measure. It worth mentioning, that range normalization is susceptible to noise, which means that if there are outliers, such a normalization will scale the normal values to a very small interval near zero, which can cause inconvenience as most of the data sets have outliers.

Log normalization transforms each value using a logarithm. It uses logarithms to better represent skewed data. Log normalization is helpful when values are clustered around small values with a few large values. Each value of a new matrix is calculated by subtracting its column's minimum, adding one, and then taking the logarithm of that value according to the formula:

$$x_{ij}^{*} = \log\left(x_{ij} - \min_{j} x_{ij} + 1\right).$$

Log transformation reduces impact of outliers, but does not exhibit constant mean, range or variance values across variables.

Final results can significantly differ depending on the normalization technique applied in the preprocessing phase. Despite existence of significant amount of normalization methods, those matched the data and the goal of the research must be utilized.

2.3 Dimensionality reduction

In many practical applications there are far too many attributes, or features, to be handled by learning schemes. Some of the attributes can be irrelevant or redundant. Many learning methods try to appropriately handle attributes and ignore redundant ones, however, in practice their performance and speed can often be improved by dimensionality reduction. Thus, the data must be preprocessed to select a number of attributes to use in learning. Manual selection of attributes, based on a deep understanding of the problem area and meaning of the attributes, is considered the best technique for feature selection. Nevertheless, automated methods are also widely used.

This section covers the main linear technique for dimensionality reduction – Principal Component Analysis (PCA). PCA description, algorithm and evaluation techniques are discussed.

2.3.1 Principal Component Analysis description

Principal Component Analysis (PCA) (Pearson, 1901) is a linear technique that allows projecting n -dimensional data onto a lower dimensional subspace in a way that is optimal in the least-squares sense. PCA can be understood in two different ways. First it reveals the minimum reconstruction error. That is the reconstructed data must have as little difference from the original data as possible, while keeping a fixed amount of principal components that contain the main part of information and are uncorrelated. In a typical dimensionality reduction task components, which contribute to the original data less information than a certain predefined fixed threshold (expressed in percentage of the source data explained) are discarded. It can also be viewed as a technique that maximizes variances of the original vector projections on a new rotated coordinate system, where these projections are uncorrelated. The components that have very low variance are omitted in the latter case.

The starting point for PCA is a random vector x with n elements, which is described by a sample $x(1), \dots, x(T)$. No explicit assumptions on the probability density of the vectors are made in PCA, as long as the first- and second-order statistics are known or can be estimated from the sample. Also, no generative model is assumed for the vector x. It is essential in the PCA that the elements are mutually correlated, so that there is some redundancy in x making the compression possible. If the elements are independent, nothing can be achieved by PCA.

In the PCA transformation, the vector x is first centered by subtracting its mean:

$$x \leftarrow x - E\{x\}\,.$$

In practice such a mean is estimated from the available sample $x(1), \dots, x(T)$ according to the following formula:

$$E\{x\} = \frac{1}{T}\sum_{i=1}^{T} x(i)\,.$$

Here T is the number of available realizations of vector x. Further let us assume that the centering has been done and, thus $E\{x\} = 0$. Next, x is linearly transformed to another vector y with m elements, where $m < n$, so that redundancy induced by the correlation, or dependency of variables, is removed. This is done by finding a rotated orthogonal coordinate system such that the elements of x in the new coordinates become uncorrelated. At the same time, the variances of the projections of x on the new coordinate axes are maximized so that the first axis corresponds to the maximal variance, the second axis corresponds to the maximal variance in the direction orthogonal to the first axis, and so on.

2.3.2 Principal Component Analysis algorithm

A variant of a step by step algorithm of the PCA is described below.

1. Compute n -dimensional mean $\mu = \frac{1}{N} \cdot \sum_i x(i)$.

2. Compute $n \times n$ covariance matrix $C_x^{k,l} = \frac{1}{N} \cdot \sum_i (x_k(i) - \mu_k) \cdot (x_l(i) - \mu_l)$, for all $k, l = \overline{1,n}$.

3. Compute eigenvectors and eigenvalues of C_x by solving a set of n equations: $C_x \cdot w_k = \lambda_k \cdot w_k$, subject to $\| w\| = 1$, $w_k \perp w_l$.

4. Choose m largest eigenvalues. m is the inherent dimensionality of the subspace governing the data and $(n - m)$ dimensions generally contain noise.

5. Form an $n \times m$ matrix W with m columns of eigenvectors in the order determined by the largest eigenvalues.

6. The representation of data by principal components consists in projecting data into m -dimensional subspace by $y = W^T \cdot (x - \mu)$.

Also there are Neural Network implementations of the PCA available.

2.3.3 Principal Component Analysis evaluation

To perform successful feature extraction and an assignment of principal components to features, producing "best" features, there is a need to introduce an appropriate principal component evaluation criterion. This criterion must be chosen in accordance with the eventual goal of the research process (feature extraction, dimensionality reduction, etc.), when seeking for the optimal solution of the problem (e.g., best correlation of features). Optimal solution (the set of fixed parameters – fixed way of assigning principal components to features) gives the best fitness of the resulting function being optimized (e.g., maximization of feature correlations).

Selection of the most important principal components can be done in different ways. The simplest way to define the principal component evaluation criteria is to introduce some ordering of the set of principal components. The ordering must express the importance of the principal component for a particular feature assignment. There are several options for ordering:

1. Following the effect on data as the result of PCA application, it is natural for PCA to use the ordering of the principal components by their vari-

ance (variance of the data projection to principal directions). In such a way principal components having greater variance are counted as more important, containing most information about the data distribution, and principal components having small variances are treated as "noisy" irrelevant components. However, this interpretation of the importance of principal components suits well only for the dimensionality reduction task. For other purposes, principal components having small variances may be still useful.

2. Another way to sort out principal components properly is to estimate the distributions of principal components. Since it might be known in advance (by the experience or conditions and constraints of a concrete task) that certain distributions are critical or otherwise "noisy", someone might wish to preserve them or get rid of them. In this case ordering criteria will be, actually, some measures of closeness to a specific distribution, or measures of some properties of distributions. Following this idea, principal components will be sorted by their resembling properties with the concrete distribution.

3. The third way of ordering is based on the experience of the researcher. If the domain of investigation is well-known by the researcher, he can select proper principal components and make the assignment of them to features properly based on his knowledge and subjective opinion.

3 METHODS DESCRIPTION

3.1 Supervised Self- Organizing Maps

3.1.1 SOM method description

The Self-Organizing Maps (SOM) (Kohonen, 2001) was introduced by T. Kohonen and has gained success in many application fields. Traditionally the Self Organizing Maps (SOM) is regarded as unsupervised classification and clusterization. Supervised SOM arose later, when during attempts of phonemes recognition from natural speech it turned out that the class separation and thus also classification accuracy could be improved by significant amount, if information about the class-identity were taken into account in the learning phase.

Unsupervised methods are in a disadvantage against the supervised methods as they do not use the class information of the training samples at all. On the other hand, unsupervised methods can be used even without the class labels. In real life the labels are rarely available.

SOM is a well- known neural model and is popular in areas, which require visualization and dimensionality reduction of large, high dimensional data sets. The SOM was developed in order to assist in identifying clusters in multidimensional datasets. Also, the SOM has a capability to generalize. Generalization capability means that the network can recognize or characterize inputs it has never encountered before. A new input is assimilated with the map unit it is mapped to.

The SOM algorithm is based on unsupervised, competitive learning. The idea of the method is that the SOM projects the dataset onto a q-dimensional plane where often q equals to two. In other words, it provides a topology preserving mapping from the high dimensional space to map units. Map units, or neurons, usually form a two-dimensional grid and thus the mapping is a mapping from high dimensional space onto a plane. Figure 2 below provides a general graphical representation of the SOM architecture.

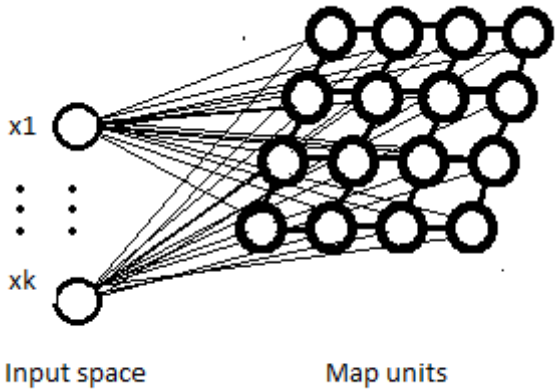


FIGURE 2 A simple Kohonen network

The SOM architecture consists of two layers:

- The input layer
- The Kohonen (map) layer

The neurons, or nodes, in the Kohonen layer are arranged in one- or two-dimensional grid. Each neuron in the input layer has a feed-forward connection to each neuron in the Kohonen layer. There are no lateral connections between nodes within the grid. The property of topology preserving means that the mapping preserves the relative distance between the points. Points that are near each other in the input space are mapped to nearby map units in the SOM and thus "similar" in the original multidimensional space data points are mapped onto nearby areas of the q-dimensional output space (Hagenbuchner and Tsoi, 2004).

3.1.2 SOM algorithm description

The Self-Organizing process is based on the following major elements.

1. **Initialization.** All connection weights between input units and neurons in the Kohonen layer are initialized with small random values.

2. **Competition.** Each unit of the input space $\mathbf{x}_i = (x_{i1}, \dots, x_{iK}),\ i = \overline{1, N}$ is mapped to a position j in a pre-defined grid of fixed dimension (Kohonen layer). Considering $\mathbf{w}_j = (w_{j1}, \dots, w_{jK}),\ j = \overline{1, L}$ is the weights vector between the i^{th} input unit and j^{th} neuron in the Kohonen layer ($\mathbf{w}_j$ is called input prototype), the discriminant function can be defined as:

$$d_{ij} = \sum_{k=1}^{K} (x_{ik} - w_{jk})^2 .$$

A neuron with the smallest value of the discriminant function is declared the winning neuron (Best Matching Unit; BMU) and can be defined as follows:

$$BMU_i = \underset{j}{\operatorname{argmin}}\, d_{ij} .$$

3. **Cooperation.** The winning neuron determines the spatial location of a topological neighbourhood of excited neurons. A neighborhood function is defined based on the distance between neurons, for example, it can be Gaussian neighborhood function:

$$h_{\mathbf{u}_i,\mathbf{u}_{BMU}} = exp\left(-\frac{\|\mathbf{u}_i - \mathbf{u}_{BMU}\|_2^2}{2\sigma^2}\right),$$

where $\mathbf{u}_i$ is a vector containing coordinates of i-th unit in the map, $\mathbf{u}_{BMU}$ is a vector containing coordinates of BMU, σ decreases with time.

4. **Adaptation.** Weight adaptation occurs interactively after each input presentation. The winning neuron modifies its input prototype $\mathbf{w}_{BMU}$, which brings it closer to the input $\mathbf{x}$ and causes weight adaptation of all the neighboring neurons via the function h (Buessler et al., 2002).

In order to make the SOM supervised, the input vectors were formed of two parts x_1 and x_2. The first part was an n-dimensional codebook vector, which is an input vector of the unsupervised SOM. The second part was a unit vector with its components assigned to one of the possible classes, which were taken into account. Then the final vector $x = [x_1^T, x_2^T]$ is used as input to the SOM. Since x_2 is the same for all vectors of the same class and different for different classes, clustering of the vectors x along with classes leads to improved class separation. This method explicitly uses class labels of the instances in the learning phase.

The unsupervised SOM constructs a topology perceiving representation of the statistical distribution of all input data. The supervised SOM tunes this representation to discriminate better pattern classes.

3.2 Naïve Bayes

Bayesian classification is named after T. Bayes, who proposed the Bayes Theorem (Bayes et al., 1940). Naïve Bayes is a supervised learning method and at the

same time it is a statistical method for classification. Naïve Bayes classifiers are widely used in different areas dat. They are well known among the most successful algorithms for learning to classify text documents, spam filtering and etc.

A Naïve Bayes classifier is a probabilistic classifier, which is based on estimating conditional probabilities of an observation variable from the training data set and using them for classification of new data instances. The main idea of Naïve Bayes classifiers is as follows. In classification learning, each instance is described by a vector of attribute values and its class can take any value from some predefined set of values. A set of training instances with their class labels, the training dataset, is provided, and a new instance is presented. The learner is asked to predict the class for this new instance according to the evidence provided by the training dataset.

Considering D is given observed data, H is a set of candidate hypothesis, $h \in H$ is a particular hypothesis, the Bayes' theorem can be formulated in the following way:

$$P\ (h\ |D) = \frac{P\ (h)\ P\ (D\ |h)}{P(D)}.$$

Here is a list of definitions related to the formula above.

- $P(h)$ represents prior probability of hypothesis h and reflects any background knowledge about the chance that h is a correct hypothesis (before having observed the data).

- $P(D)$ represents prior probability of data D and reflects the probability that data D will be observed given no knowledge about which hypothesis h holds.

- $P(D|h)$ represents conditional probability of observation D and denotes the probability of observing data D given that hypothesis h holds.

- $P(h|D)$ is posterior probability of h and represents the probability that h holds given the observed data D.

The goal of the Bayes' theorem is to determine the most probable hypothesis, given the data D and initial knowledge about prior probabilities of various hypotheses in H. To find the most probable hypothesis h from a set of candidate hypotheses H, given the training data D, a maximum a posteriory (MAP) hypothesis can be calculated, according to the formula:

$$h_{MAP} = arg \max_{h \in H} P\ (h\ |D)$$

$$= arg \max_{h \in H} \frac{P\ (h)\ P\ (D\ |h)}{P(D)}$$

$$= arg \max_{h \in H} P\ (D\ |h) P(h)\ .$$

If prior probabilities $P(h)$ of every hypothesis from H are equal, then the formula above can be simplified through choosing maximum likelihood (ML) of the data D given h:

$$h_{ML} = arg \max_{h \in H} P\ (D\ |h).$$

In classification tasks Bayes' theorem can be used to calculate the probability of each class given the instance x:

$$P\ (C\ =\ c\ |X\ =\ x) = \frac{P\ (C\ =\ c)P\ (X\ =\ x\ |C\ =\ c)}{P(X\ =\ x)}.$$

Here C is a random variable denoting the class of an instance, c is a particular class label, $X < X_1, X_2, \cdots, X_k >$ is a vector of random variables denoting the observed attribute values (an instance), $x < x_1, x_2, \cdots, x_k >$ is a particular observed attribute value vector (a particular instance).

Figure 3 below shows graphical representation of Naïve Bayes model, where attribute nodes are independent from each other given the class label C.

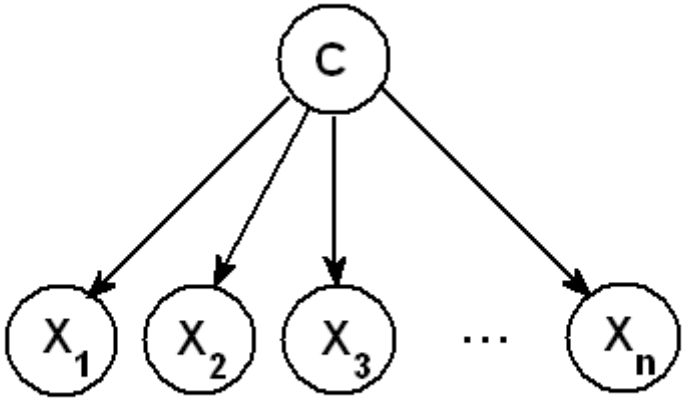


FIGURE 3 Graphical model of a Naïve Bayes classifier

Bayes' rule decomposes the computation of posterior probability into the computation of likelihood and prior probability. Expected error can be minimized by choosing the class with the highest probability as the class of the instance x. Because the probabilities needed for the calculation are not known, it is necessary to estimate them from the training dataset.

Bayesian classification calculates explicit probabilities for hypothesis and it is robust to noise in input data. Naïve Bayes classifiers are simple, efficient and robust to noise and irrelevant attributes. One defect, however, is that Naïve Bayes classifiers utilize an assumption that the attributes are conditionally independent of each other given the class. Although this assumption is often violated in the real world, the classification performance of Naïve Bayes classifiers

is still surprisingly good for many classification tasks, compared with other more complex classifiers. According to (Domingos et al., 1997), this can be explained by the fact that classification estimation is only a function of the sign (in binary cases) of the function estimation, the classification accuracy can remain high even while function approximation is poor and by considering the fact that we are not interested in conditional probability $P(h|D)$, but merely in calculating the most likely class.

Another thing that can go wrong with Naïve Bayes is that a particular attribute value might not be presented in the training set in combination with every class value. This puts zero probability on the attribute value given this class and because the other probabilities are multiplied by this one, the final probability for this class will be zero. Unfortunately, a zero estimate can create significant problems for classification process. This however can be solved by minor adjustments to the method of calculating probabilities from frequencies. To overcome this problem, smoothing procedures are usually used to avoid zero probability estimates in practice. One of the standard techniques is called Laplace smoothing, which can be done via the formula:

$$P(x_i|C) = \frac{count(x_i|C) + \mu}{N_C + |x|\mu}.$$

Here $x = (x_1, \dots, x_n)$ is a feature vector, $count(x_i|C)$ is the number of times the feature x_i appears in the class C , N_C is the number of values that the class C can take, $|x| = n$ is size of the feature vector x, and μ is a smoothing parameter. A special case of Laplace smoothing is "add one" smoothing, which is obtained by setting $\mu = 1$. Nevertheless, there is no particular reason for adding one, instead, a small constant can be used. Such smoothing parameter provides a weight, which determines influence of the a-priory values for all possible attribute values. The larger the smoothing parameter value, the higher the importance of priors compared with the new evidence coming in from the training set. On the other hand, small values of the smoothing parameter give them less influence.

3.3 Support Vector Machine

3.3.1 SVM method description

Support Vector Machine (Vapnik, 1995) is a kernel based method, which is used for solving pattern classification and function approximation problems. In pattern classification problems a classifier is constructed so, that the objects are classified correctly with a high accuracy. Input to such a classifier is called features, which represent each class well or are determined so that data from different classes are well separated in the input space (Abe, 2010).

The idea of SVM is minimization of an upper bound of generalization error through maximizing the margin between the data and separating hyper-

plane. Margin is the sum of distances from the separating hyperplane to the closest data points belonging to each class. Such closest data points are called Support Vectors.

The basic idea of linear SVM is to construct a hyperplane as the decision plane, which would separate positive and negative classes with the largest margin. If the classes are linearly separable then the optimal separating hyperplane would be the one, which gives the smallest generalization error among all possible hyperplanes. Such a hyperplane would be the one, which has the maximum margin between the classes. In practice two classes are not fully separable. Hence in order to find the optimal hyperplane, an optimization problem must be solved.

In general case SVM constructs a hyperplane or a set of hyperplanes in a high- or infinite- dimensional space, which can be used for classification. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training data point of any class, since in general the larger the margin the lower the generalization error of the classifier.

Whereas the original problem may be stated in a finite dimensional space, it often happens that the sets to discriminate cannot be separated in that space. For this reason, the original finite-dimensional space is mapped onto a much higher dimensional space, presumably making the separation easier in that space. The hyperplanes in the higher-dimensional space are defined as a set of points whose inner product with a vector in that space is constant. The vectors defining the hyperplanes can be chosen to be linear combinations with parameters of images of feature vectors that occur in the data base. To keep the computational load reasonable, the mappings used by SVM schemes are designed to ensure that dot products may be computed easily in terms of the variables in the original space, by defining them in terms of a kernel function selected to suit the problem.

SVM has been successfully applied in many areas such as object detection and recognition, text detection and categorization, information and image retrieval, prediction (Tay and Cao, 2002; Mager et al., 2008; Byun and Lee, 2002), etc.

3.3.2 SVM tools description

In this research LIBSVM toolbox (Chang and Lin, 2011) has been chosen as a tool for applying the SVM. LIBSVM is a library for SVM. It is currently one of the most widely used SVM software. A typical use of LIBSVM consists of two steps: (1) a model prototype is trained to obtain a model and (2) the model obtained is used to predict information of a test set.

The classification algorithm is organized in the following way. Given training vectors $x_i \in R^n, i = 1, \dots, l$ with labels belonging to two classes and indicator vector $y \in R^l$ such that $y_i \in \{1, -1\}$, the following optimization problem is being solved:

$$\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i$$

subject to

$$y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i ,$$
$$\xi_i \geq 0, i = 1, .., l .$$

Here $\phi(x_i)$ maps x_i into a higher dimensional space and $C > 0$ is the regularization parameter. The usual way to cope with the possible high dimensionality of the vector variable w is considering the following dual problem:

$$\min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - e^T \alpha$$

subject to

$$y^T \alpha = 0 ,$$
$$0 \leq \alpha_i \leq C, i = 1, \dots, l .$$

In the formulas above $e = [1, \dots, 1]^T$ is the vector of all ones, Q is $l \times l$ positive semi-definite matrix, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$ and $K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j)$ is the kernel function.

After the above dual problem is solved, from the primal-dual relationships, the optimal w is obtained in the following way:

$$w = \sum_{i=1}^{l} y_i \alpha_i \phi(x_i)$$

with the decision function

$$sgn(w^T \phi(x) + b) = sgn\left(\sum_{i=1}^{l} y_i \alpha_i K(x_i, x) + b \right).$$

Obtained values for $y_i, \alpha_i, \forall i, b$, label names, support vectors, kernel parameters are stored in the model for prediction.

After solving the optimization problem described above and obtaining a model, labels can be predicted on the test set. Classification accuracy in this case is calculated according to the following formula:

$$Accuracy = \frac{\#\text{correctly predicted data}}{\#\ total\ testing\ data} \times 100\% .$$

For classification, in the training phase LIBSVM decouples a multi-class problem to a two-class problem and calls one-class SVM several times. For multi-class classification LIBSVM implements the "one-against-one" approach (Knerr et al., 1990), which consists of constructing one SVM for each pair of

classes, which are trained to distinguish the points of one class from the points of another class.

In classification a voting strategy is used, where each binary classification is considered to be a voting. During this procedure votes are assigned to all data points, then in the end a class with maximum number of votes is designated for a data point. If two classes have identical amount of votes, then the class appeared first is chosen.

3.4 Validation and Evaluation

Validation techniques are motivated by two fundamental problems in pattern recognition: model selection and performance estimation. In real application there is a limited number of examples, which is usually smaller than wanted. One approach is to use the whole training data set to select the classifier and estimate the error rate. However, this approach has two fundamental problems: (1) the final model will most likely over-fit the training data and (2) the error rate will be under-estimated (lower than the true error rate).

Estimating the accuracy of a classifier is important for several reasons. First of all it is used for estimating future prediction accuracy of the classifier. Secondly, it is used for choosing a classifier from the given set, the best model from several available qualification models.

There are several estimation methodologies. The most commonly used methodologies are described below.

The holdout (simple split) method is an example of simple validation. In this method the data set is split into two parts: training set and test set. The training set is used to train a classifier, the test set is used to estimate the error rate of the trained classifier. This method however has some drawbacks. The main disadvantage of this method is that it assumes that the data in the two subsets are of the same kind (has the exact same properties) and since this is a simple random partitioning, such an assumption may not hold true (David, 2012). The limitations of this method can be overcome with a set of resampling methods, which require more computations.

There are a number of cross validation techniques, which are more advanced, comparing with the simple split.

Random subsampling cross validation is also known as Monte Carlo cross validation. The idea of this method is that the data set is randomly split into training and test subsets without replacement. For each such split the model is fit to the training data then the estimate is evaluated using test data. The results are then averaged over the splits.

The advantage of this method is that it can be repeated indefinite number of times (Maimon and Rokach, 2005). Its estimate is significantly better than the holdout estimate. Comparing to k-fold cross validation, the proportion of the training/validation (test) split does not depend on the number of folds.

The disadvantage of this method is that some examples may never be selected to the validation (test) set, when others can be selected more than ones. In other words, test sets are not independent with respect to distribution of examples (Maimon and Rokach, 2005).

In the **k-Fold cross validation** the data set is randomly split into k subsets of equal size. These subsets are mutually exclusive. Then the classification model is trained and tested k times. For each of k experiments, the model is trained on the k-1 subsets and tested on the remaining single subset. These subsets are called folds. The cross validation estimate is calculated by averaging the k individual performance measures using the following formula:

$$\mathrm{P} = \frac{1}{k}\sum_{i=1}^{k} P_i \,,$$

where P_i is a performance measure of i-th experiment.

The main advantage of the k-fold cross validation is such that all the examples in the data set are used for training and testing. In other words, it reduces the bias associated with the random sampling of the training and holdout data samples by repeating the experiment k times, each time using a separate portion of the data as holdout sample (David, 2012). The main disadvantage of this method is that the training and testing procedures must be repeated k times.

Leave one out cross validation is similar to the n-fold cross validation, where n equals to number of instances in the data set. Every time the method is trained on all but one instance and tested on the remaining instance. Thus, the experiment is repeated until all instances of the data set have been passed. The cross validation estimate is calculated in the same way through averaging the n individual performance measures, according to the formula above.

Among the advantages of this method are size of the training set and efficiency. The training performs on the maximum possible training set with greatest possible amount of data used in every case, as there is only one instance left for testing. This significantly increases the chance that the classifier is accurate. Because random sampling is not involved, leave-one-out is efficient comparing to other methods, where fold content must be randomized. Thus, the procedure is deterministic and does not require to be repeated to acquire accurate results, as the same result will be obtained every time. On the other hand, this method has high computational cost, as the learning procedure must be executed n times and therefore is not eligible for large data sets. However, it can work pretty well for a small data set. There is as well a disadvantage of this cross validation, which is it guarantees a non-stratified sample. Stratification requires having correct proportion of the examples of each class in the test set, which is impossible when the test set contains only a single example (Witten and Frank, 2005).

Bootstrapping is a statistical method for estimating, based on the procedure of sampling with replacement. The idea of this method is to sample a data set with replacement to for a training set. A particular bootstrapping approach

is called 0.632 bootstrap, where a data set of n instances is sampled n times with replacement in order to obtain a new data set of n instances. Some elements in the new data set can be repeated and thus there are might be elements in the original data set, which have not been picked and will be further used as test instances. The probability of a chance of a particular instance not being picked at all is calculated in the following way:

$$(1 - \frac{1}{n})^n \approx e^{-1} = 0.368\,.$$

Thus for a large enough data set, the test set will contain 36.8% of the instances and the training set will contain 63.2% of the instances. In the training set some instances will be repeated.

The disadvantage of this method is that only 63.2% of data is used in the training set. This number is quite pessimistic comparing for example to 90% in 10-fold cross validation. However an advantage of the method is that the procedure can be repeated any number of times, allowing statistical tests (David, 2012; Witten and Frank, 2005).

4 HANDLING DATA

4.1 Data description

The mobile phone data was collected on a 24/7 basis over several months. In addition to mobile data, the ground truth data was also collected. These data are semantic labels created by the users related to places that were visited frequently during the data collection period. In Figure 4 below a schematic representation of the data collection process can be seen.

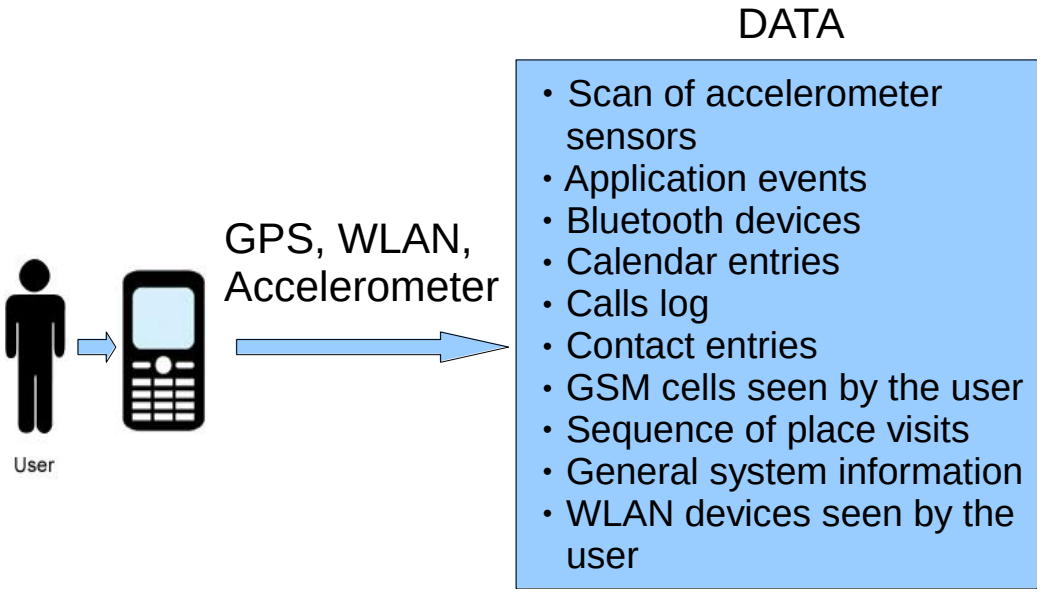


FIGURE 4 Data collection

A set of data files has been obtained from the users' mobile phones. These files contain information related to mobile phone and its activity, including information about accelerometer sensors scan data, application events, Bluetooth devices, GSM cells and WLAN devices seen by the user, calendar and contacts entries, call log, media and how it was played, running processes, mobile phone

system information and the sequence of place visits, which are longer than specific period of time. A part of information from these files has been extracted and converted to a data set used in the current research. The raw location data was transformed into a symbolic space, which excludes actual geographic coordinates. In order to do so, first, the visited placed were detected, then the sequence of visits to checked-in places was mapped to the sequence of coordinates. All places are user-specific and each place corresponds to a circle with a radius of 100 meters. Places are ordered by the time of the first visit and thus the obtained dataset is continuous. In order to keep user privacy many fields have been anonymized (Laurila et al., 2012).

Along with the data set itself, a list of place labels has been provided as input for the current research. It is a file containing three columns: user id number, place id number and place label. All places are user-specific, however place labels are common for all the users from the data set. There are ten possible values for place labels, which are listed in the Table 1 below.

TABLE 1 Place labels

Place label	**Place label description**
1	Home
2	Home of a friend, relative or colleague
3	Workplace/school
4	Location related to transportation
5	The workplace/school of a friend, relative or colleague
6	Place for outdoor sports
7	Place for indoor sports
8	Restaurant or bar
9	Shop or shopping center
10	Holiday resort or vacation spot

In the current research a data set, consisting of mobile phone data collected from 80 users during quite long period of time, has been used. The training set and the test set have been built based on the data set. Further the training set has been used for training and parameters validation, the test set has been used for performance evaluation.

The following challenges can be emphasizes with regard to the data in the obtained data sets:

- Some data in the data set are wrong or missing. Such incorrect data must be handled properly in order to make results correct.

- Some files are huge, and thus needed to be processed by parts. This procedure costs significant amount of resources and time.

- All data have different types. In order to make it possible applying data mining methods, the data need to be represented as one type as well as the appropriate normalization and scaling must be performed.

- As there is a limited amount of instances, which becomes even less after removing all incorrect data, the results might not be accurate enough.

- Labels are unbalanced. Some labels significantly dominate upon others, which makes the learning of the algorithms less accurate for the labels rarely appeared in the dataset.

4.2 Data postprocessing

Data postprocessing usually includes various steps. Among these steps are interpretation, evaluation, and integration of the results. This section is devoted to the evaluation techniques and integration procedure.

4.2.1 Evaluation

Evaluation is an important step in data mining. There are many different techniques for methods' performance evaluation. Some of them have been described in section 3.4. However, comparing performance of different methods on a given problem is another nontrivial issue. To ensure that differences in methods' performance are not caused by chance effect, statistical tests are needed. For a problem of classification, the ability to classify test instances is being predicted. But, different classification methods use different techniques for predicting this ability. Thus, for example some methods are based on predicting class probability, instead of a class itself, while others involve predicting numerical rather than nominal values. In each case different performance evaluation methods are need. A set of evaluation techniques to be used for comparing performance of different methods are described in this subsection.

Error rate is a simple evaluation measure for classification tasks, which measures overall performance of a classifier. Thus, a class predicted by classifier is counted as "success" if prediction is correct and "error" otherwise. The error rate is calculated as proportion of errors made over the whole test set.

In classification tasks the measures true positives, true negatives, false positives, and false negatives are widely used for calculating more advanced performance evaluation measures. True positives (TP) and true negatives (TN) are correct classifications. False positive (FP) occurs when a class is incorrectly predicted as positive, and false negative (FN) occurs when a class is incorrectly predicted as negative.

Recall-precision evaluation is based on measure of relevance. Precision is a fraction of retrieved instances that are relevant. Recall is a fraction of relevant instances that are retrieved. They can be calculated using the formulas:

$$Precision = \frac{TP}{TP + FP},$$

$$Recall = \frac{TP}{TP + FN}.$$

High recall means that an algorithm returned most of the relevant results. High precision means that an algorithm returned more relevant results than irrelevant.

F-measure (Rijsbergen, 1979) is also used in information retrieval. In a traditionally F-measure precision and recall are evenly weighted. Such a measure is called F1-score and can be calculated via the formula:

$$\mathrm{F}_1 = 2\,\frac{\mathrm{precision}\ \times \mathrm{recall}}{precision + recall}.$$

However in a more general case, a weight can be assigned to put more emphasis either to precision or recall. In this case the formula for calculating the measure can be re-written in the following way:

$$\mathrm{F}_\beta = (1 + \beta^2)\,\frac{\mathrm{precision}\ \times \mathrm{recall}}{\beta^2 \times precision + recall},$$

where $\beta \geq 0$ is a weight constant.

There are number of different techniques for performance evaluation, however only those used in this Master's thesis have been described.

4.2.2 Integration

A list of place labels, provided as input for the current research, is a file containing three columns: user id number, place id number and place label. All records in the file are unique and can be used for evaluation of the classification results. This file has size of 336, which means that overall in the data set there are 336 unique pairs corresponding to user id number and place id number. Although one label can correspond to different combinations of user id and place id, the mapping of a pair "user id – place id" to a specific label must be unique.

Thus, for accurate evaluation of the predicted via classification models labels, an additional transformation technique is required. This transformation must convert the labels obtained after applying classification model to the whole data set, into a file having the format "user id – place id – label id", suitable for evaluation.

5 EMPIRICAL PART

5.1 Planning of the machine learning pipeline

During the implementation of the classification approach, several important practical concerns must be taken into account.

Input selection and dimensionality reduction. To ensure the optimal performance the features must be carefully selected. Some methods, like SOM do not depend on whether the features correlate or not, as they perform this analysis themselves. However, there are methods, like SVM or Neural Networks, which require that the input features should be as much independent, as possible. For this purpose dimensionality reduction techniques must be applies. These concerns have been discussed in details in Chapter 2.

Generalization. It might appear that a model trained on a data set shows poor prediction results on new data. This phenomenon is called overfitting and it is related to the ability of the model to generalize. There are a number of techniques, which are used for assessing such model's ability. Some of them have been listed and described in section 3.4.

Method selection and evaluation. A method for classification must be chosen according to the data being analyzed and a problem being solved. Careful analysis of the data in preprocessing phase is capable to influence performance on the later stages. Depending on the data, the results shown by different methods can vary. There is a set of techniques to be used for methods' performance evaluation. They are discussed in subsection 4.2.1.

5.2 Application of the methods

This section is devoted to empirical results of the methods used during data analysis process in the research. Subsection 5.2.1 describes feature selection process and lists number of features selected during initial data analysis phase. Re-

sults of applying normalization and dimensionality reduction techniques are presented in subsections 5.2.2 and 5.2.3 correspondingly. Subsections 5.2.4-5.2.6 cover methods used for data classification as well as graphical representation of the results.

5.2.1 Feature selection

According to the information, provided in the Data Description section, a set of semantic labels related to places visited by the users has been collected. In order to successfully predict semantic labels, it is essential to extract relevant features from the data sets. To obtain needed features almost all the files from the provided data have been used. Since all the places are user-specific, extracted features are supposed to be generic for all users. 410 features have been extracted, 386 out of them are binary features related to applications.

The following features related to places have been extracted from the data.

Values related to accelerometer sensors:

- Time spent by a user in a place.
- Average value of square change of all the values divided by the number of samples (avdelta).

Values related to application events:

- Number of unique active applications, collected every second.
- Status of application. For every existent application this value equals to 1 if it is active, and to 0 otherwise.

Values related to Bluetooth devices:

- Average number of Bluetooth devices seen by a user.

Values related to calendar entries:

- Number of calendar entries added by a user, collected every second.

Values related to calls log:

- Percentage of time a user spent speaking by phone.
- Number of incoming calls, collected every second.
- Number of outcoming calls, collected every second.
- Number of incoming text messages, collected every second.
- Number of outcoming text messages, collected every second.

Values related to contact entries:

- Number of interactions with contacts from the phonebook, collected every second.
- Number of contacts added, collected every second.

Values related to GSM cells that a user has seen:

- Quality of average network signal strength (from 0 to 7).
- Average network signal strength (in dBm).

Values related to media files played by a user:

- Number of media files played.
- Percentage of the time media files were playing.

Values related to system information about the phone:

- Average battery level. All records related to invalid battery level namely having battery level higher than 100%, have been removed.
- Percentage of time a phone was charging.
- Percentage of time a phone was inactive.
- Dominating type of ring.
- Average amount of free RAM.

Values related to WLAN devices seen by a user:

- Average received WLAN signal level.
- Dominating security mode.
- Dominating operational mode.

Finally two data sets containing 32903 instances each have been obtained:

1) D1 (389 features related to applications). Each feature is a binary representation of a particular. One means that the application is active, and zero otherwise, at a particular time interval. All records are ordered by time.

2) D2 (24 features not related to applications). This data set contains only numerical features. All records are ordered by time.

Both data sets have a common problem, which is place labels are not equally distributed among all the data, in other words, labels are unbalanced. Thus, place labels 1, 2 and 3 significantly dominate upon others, whereas there are only a few records related to places with label 10. Other place labels are

more or less equally distributed among the remaining data, varying around 1%. Table 2 below represents relative amount of records in the data set, placed in descending order by their percentage ration.

TABLE 2 Place labels ordered by presence in the data set (in %)

Place ID	%
1	49.59
3	39.25
2	6.18
7	1.34
6	0.93
9	0.92
8	0.87
5	0.57
4	0.33
10	0.02

In order to treat the records more or less similarly during training phase, the data sets have been re-organized. On the basis of the data sets D1 and D2, two reduced data sets have been formed in the following way. All records related to dominating place labels 1, 2 and 3 have been separated from the original data sets D1 and D2, then they have been shuffled. Only 1.5% of records related to every label from the separated data set above have been chosen and added to new data sets D1_reduced and D2_reduced. These two reduced data sets are relatively balanced with regard to place labels, and have been used in the training phase of the methods Supervised SOM, Naïve Bayes and SVM as further described in the related sections.

Further, data set D1 (D1_reduced) will be used for the statistical Naïve Bayes method, and data set D2 (D2_reduced) for the Supervised SOM and SVM. It can be explained by the fact that data set D1 is categorical and thus it is easier to perform parameters evaluation in the Naïve Bayes classification using this data set. Moreover, quick tests have shown that other methods do not work well on this data set and thus will not be covered in this Master's thesis.

5.2.2 Normalization

In the dataset D2 there are variables, which are much greater than others. These variables will mask others. Moreover, there might be outliers. It is reasonable to apply one of the standardization methods.

Z-score standardization has been chosen, where each value of the obtained numeric matrix is calculated from values of the current matrix as follows:

$$x_{ij}^{*} = \frac{x_{ij} - \mu_j}{\sigma_j}.$$

Here σ_j is mean value of the j-th column and μ_j is the j-th variable standard deviation. Each column of the new matrix has a mean of zero and a variance of one, and affects the result in a similar way.

The data set D1, containing features related to application, does not require normalization, as it contains only zeros and ones.

5.2.3 Principal Component Analysis

Data set D2 containing 24 features not related to applications has been used. Firstly z_scores normalization has been applied to the data set. Secondly MATLAB princomp method has been chosen to perform PCA on the normalized data set D2. Variance explained has been calculated according to the formula:

$$variance\_explained = \frac{100 \times variances}{sum(variances)},$$

where $variances$ is a vector containing the eigenvalues of the covariance matrix of the normalized data set D2. Figure 5 below represents the results of PCA applied to the data set D2, or more precisely, principal components ordered by their variance.

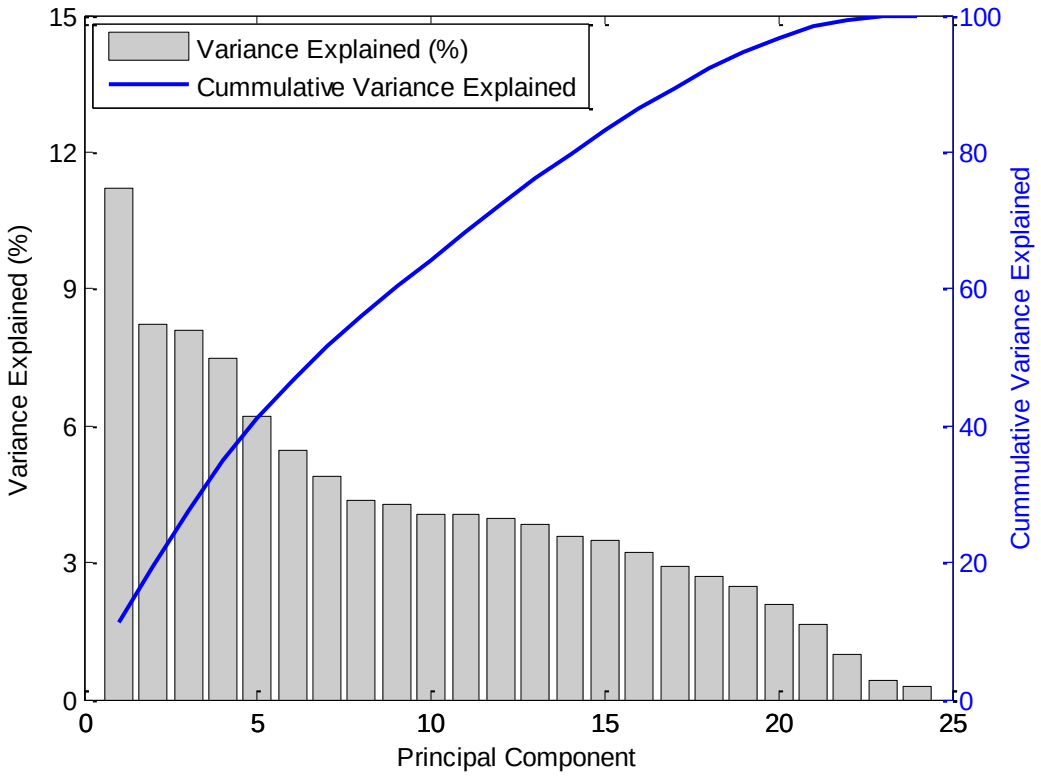


FIGURE 5 Principal components of the data set D2, ordered by their variance

According to Figure 5 above, only a few last components, which have small variance values can be treated as noisy and be candidates for removal. However their removal would not significantly reduce dimensionality of the data set, moreover current dimensionality of 24 features is sufficiently small comparing to the number of records in the data set. Thus, it has been decided to leave all the components and further analyze the whole set of features.

On the other hand, the features of the data set D1 are categorical, as they relate to applications and can have only values of zero or one, where one appears if application is active, and zero otherwise. This data set is likely to be used for the Naïve Bayes classification. Despite the Naïve Bayes works well with both categorical and continuous data, the main reason to leave this data set unchanged is that for categorical variables there is no very strong distributional assumptions such as normality (Gaussian distribution) or unimodality (meaning that the distribution curve has just a one bump in it) of the distributions. Distributions of categorical variables are also easier to understand, since it is usually sufficient to just count occurrences of values in a data matrix rather than counting sums and sums of squares and having exponent functions and so on. With categorical variables the method deals with probabilities instead of densities and sums instead of integrals (Jeffreys, 1998; Bernardo & Smith, 2007). Thus, categorical variables are much easier to be handled. Dimensionality reduction will destroy categorical property of the data set and thus it has been decided to use original data set D1 without features' transformation.

5.2.4 Naïve Bayes

For the Naïve Bayes method dataset D1 containing 386 features related to applications has been used. According to conditions of the method, features must be conditionally independent. However, Figure 6 below representing values of correlation matrix, corresponding to the data set D1, shows that there are variables, whose pair-wise correlation values are close to one. Nevertheless, this method can still show good results despite of existence of some dependency.

For training and validation of the Naïve Bayes the data set D1_reduced described in the section 5.2.1 has been used.

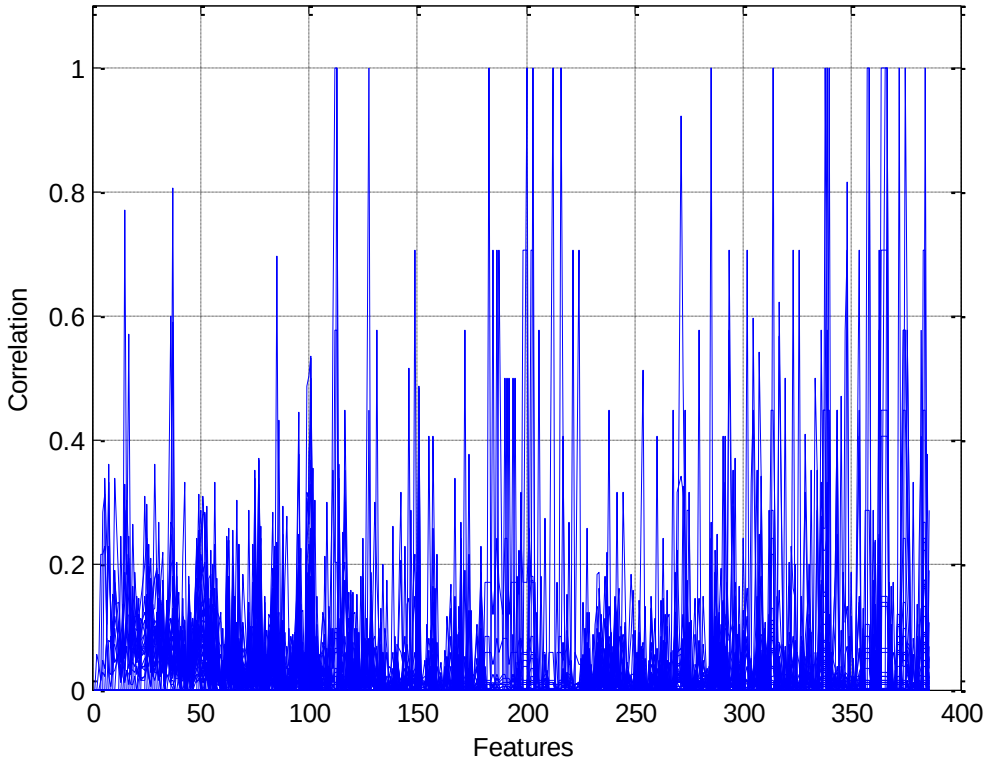
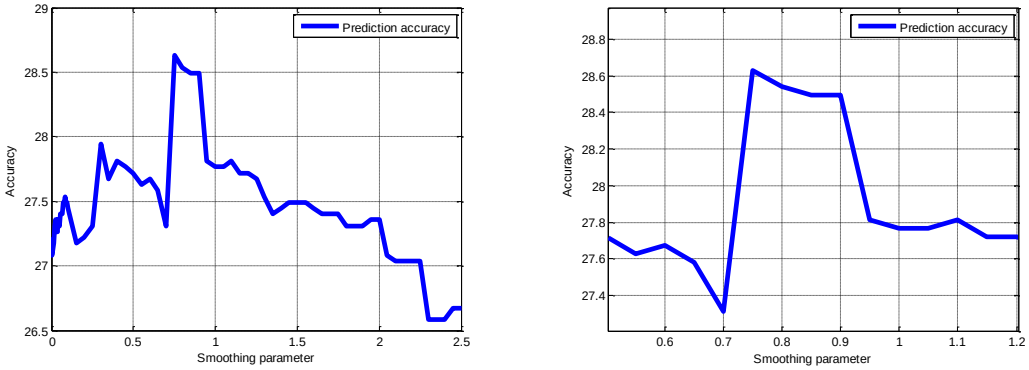


FIGURE 6 Correlation values of D1 features

Validation technique used during implementation of the Naïve Bayes is a mixture of holdout and 10-fold cross validation procedures. The data set D1_reduced has first been divided into training set (70%) and test set (30%). Then 10-fold cross validation has been performed on the training set in order to select parameters more precisely.

The Naïve Bayes has been implemented via MATLAB. This method has the only parameter, related to the smoothing procedure used to avoid zero probability estimates. Laplace smoothing with the parameter value of one is a classical choice, however, smoothing parameter can be any constant small enough. Thus, as a set of possible values for the smoothing parameter, the constants from the interval $[0, 5]$ have been tested. A number of the method iterations have been performed in order to determine most accurate parameter value for the smoothing procedure.

Figure 7 displays curves, corresponding to accuracy the method gained via the respective smoothing parameter value. In Figure 7a it can be seen that the method's accuracy first increases then decreases with increasing the smoothing parameter value, reaching its maximum in the interval [0.5,1]. Figure 7b is an enlarged part of Figure 7a. It is clearly visible that the optimal smoothing parameter value, corresponding to the maximal method accuracy, is 0.75.



(a) (b)

FIGURE 7 Results of the smoothing parameter validation for the Naïve Bayes (a) original size, (b) enlarged size

The final model has been trained on the whole training set using the selected parameters and tested on the test set. The classification accuracy of the model obtained can be seen in figures below.

Figure 8 represents original labels in blue, correctly classified labels in green and misclassified labels in red. From this figure it can be seen, what labels have been mixed up during classification. The method worked quite well for the labels 1, 2, 3 and 9. There is a small amount of correctly classified data for the labels 6 and 7. At the same time, the labels 4, 5, 8 and 10 have not been correctly classified at all.

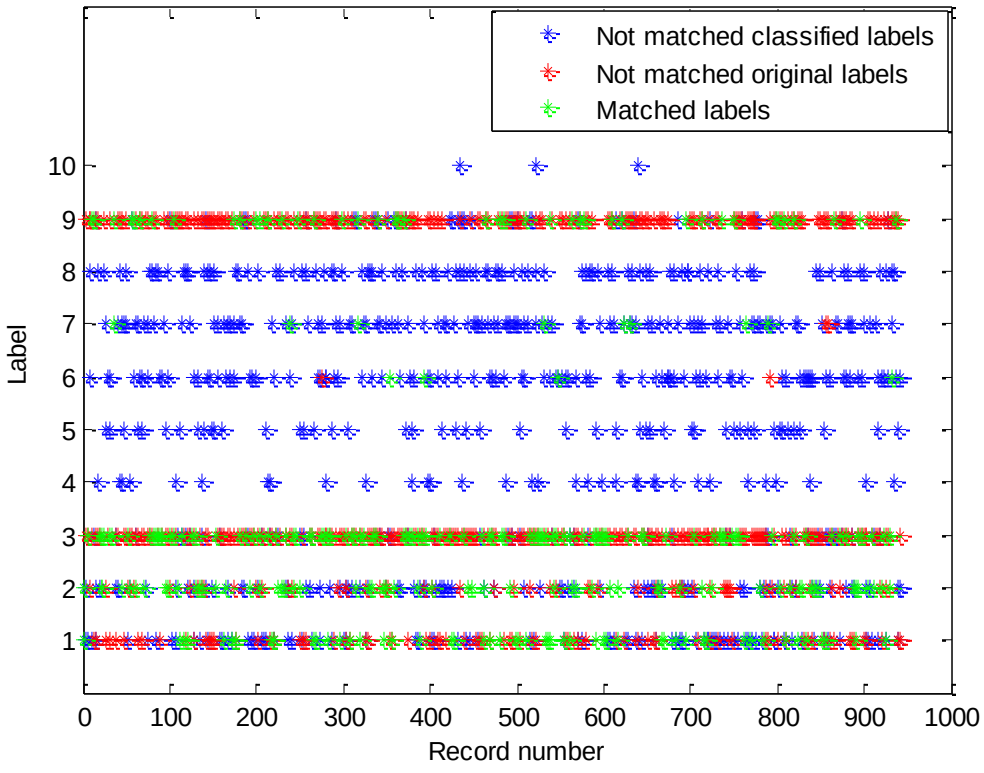


FIGURE 8 Results of the Naïve Bayes classification for the dataset D1_reduced

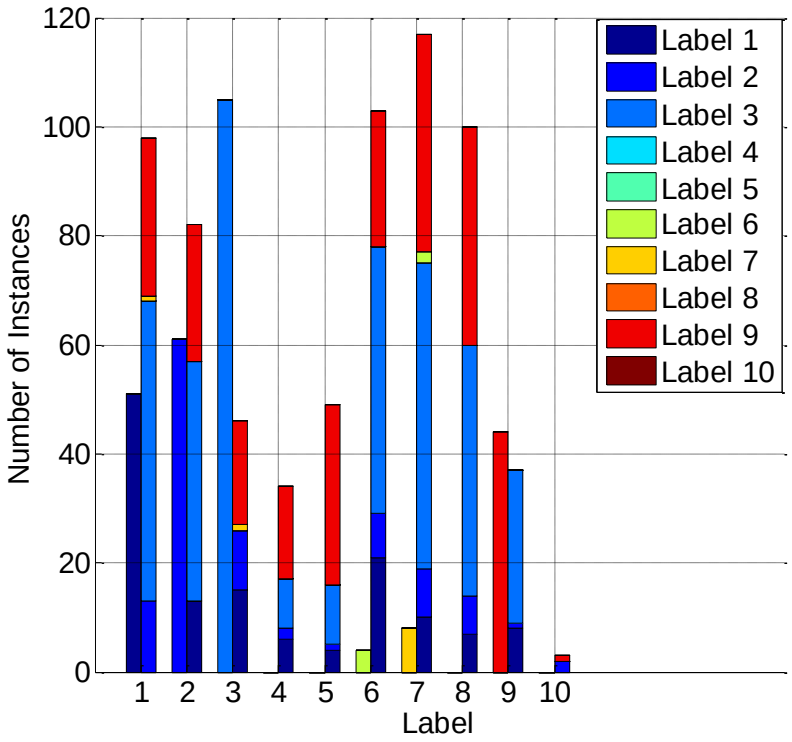


FIGURE 9 Number of correctly and incorrectly Naïve Bayes classified data for the data set D1_reduced, grouped by labels

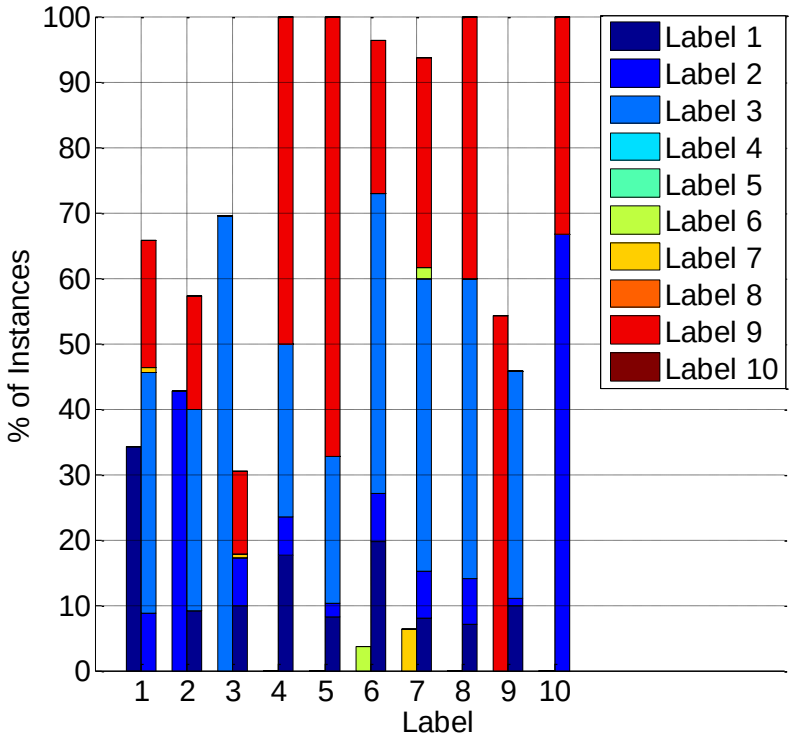


FIGURE 10 Percentage of correctly and incorrectly Naïve Bayes classified data for the data set D1_reduced, grouped by labels

Figures 9, 10 above represent amount of correctly predicted and misclassified labels in cumulative bar format. Figure 9 presents absolute number of labels and Figure 10 presents their percentage. The first bar in each group corresponds to correctly predicted labels. The second bar relates to misclassified data. It represents total amount or percentage of misclassified data as well as the number or percentage grouped by labels. Each color stripe corresponds to a particular label.

Figures 9, 10 show that the best classification accuracy, which is 70 %, belongs to place label 3, followed by labels 9 (46%), 2 (43%) and 1 (34%). The majority of misclassified labels have gone to the classes 3 and 9, followed by the classes 2 and 1. This means that this method separates well the classes 1, 2, 3 and 9 from each other, but does not distinguish those from the remaining classes 4, 5, 6, 7, 8 and 10. This result might be caused by the fact, that the classes 1, 2, 3, 9 are dominating comparing to the others. Also there is no strong evidence of distinctive sets of applications characterizing particular class of labels.

5.2.5 Support Vector Machine

Data set D2 containing 24 features not related to applications has been used for SVM. The data set D2_reduced described in the section 5.2.1 has been used in the training and validation phases.

Validation technique used along with the SVM is a mixture of holdout and 10-fold cross validation. For this purpose the data set D2_reduced has first been divided into 2 parts: training set (70%) and test set (30%). This step is important as there is no separate test set and it would be good to see how the classifying model behaves on new data. At the same time, the result of classification can be different, depending not only on parameters, but also data it is trained on. Because of this, 10-fold cross validation has been performed on the training set.

The effectiveness of SVM depends on the selection of kernel, the kernel parameters, and soft margin parameter C. There is no straightforward rule for how to choose the kernel and parameters values, however it is recommended to use linear kernel when there is a large number of features and small number of examples in the data set. If number of features is small and number of examples is large then Gaussian kernel is preferable. Polynomial kernel is rarely used in experiments (NG, 2012). A common choice of the kernel for SVM is Gaussian kernel. Moreover, in the data set D2_reduced the number of features is 24, which is rather small and the number of examples is 3139, which is quite a large number. Because of these two reasons, the Gaussian kernel has been chosen.

The kernel has a single parameter γ. Best combination of C and γ can be selected by a grid search, which can start for example from the following values:

$$C = \in \{2^{-5}, 2^{-3}, \dots, 2^{15}\}, \;\; \gamma \in \{2^{-15}, 2^{-13}, \dots, 2^{3}\}.$$

Each combination of parameter values has been checked using 10-fold cross validation. For every combination of parameters values a model has been

trained ten times and the accuracy was calculated through averaging the values obtained after ten iterations. For visualization, only limited intervals of parameters values have been selected. The results of validation for parameters $C = \in [150, 350]$, $\gamma \in [0.00003, 0.5]$ can be seen in Figure 11 below, where the vertical axis corresponds to accuracy and left and right axes correspond to the parameters γ and C correspondingly.

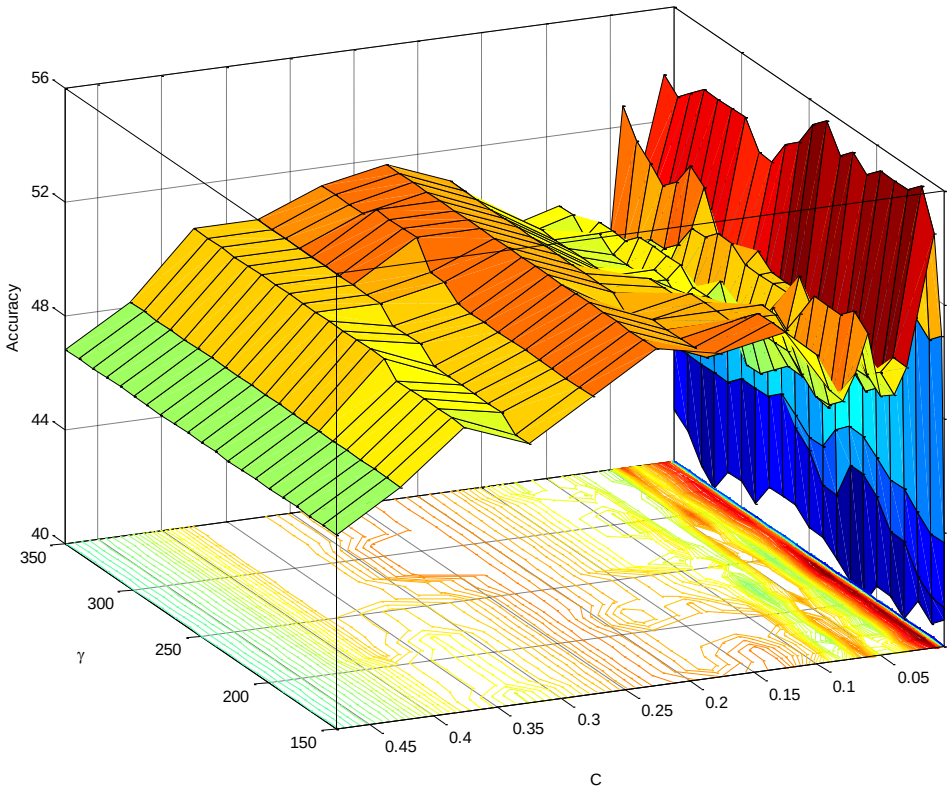


FIGURE 11 Results of grid search for the SVM

The meshed surface above is in different colors. Color palette varies from blue to red tints. Blue palette corresponds to the lowest accuracy values, and red palette corresponds to the highest accuracy values. Thus it can be seen that accuracy is more or less constant for the values $C \in [150, 350]$, however it is slightly higher in the interval $C \in [150, 250]$, reaching its maximum at $C \approx 256$. On the other hand, for the other parameter there is a clearly dominating ue $\gamma \approx 0.0625$, which in combination with $C \approx 256$ provides the highest model accuracy around 55%.

Thus, the parameters with best cross-validation accuracy are $C = 2^8$ and $\gamma = 2^{-4}$. The final model has been trained on the whole training set using the selected parameters. The model obtained has been tested on the test set, which consists of 30% from the data set D2_reduced. The classification accuracy can be seen in figures below.

Figure 12 represents original labels in blue, correctly classified labels in green and misclassified labels in red. Looking at this figure it is possible to see, what labels have been mixed up during classification. The method worked sufficiently well for almost all label classes, except for the class 10, which might be due to a relatively small number of instances of this class in the data set.

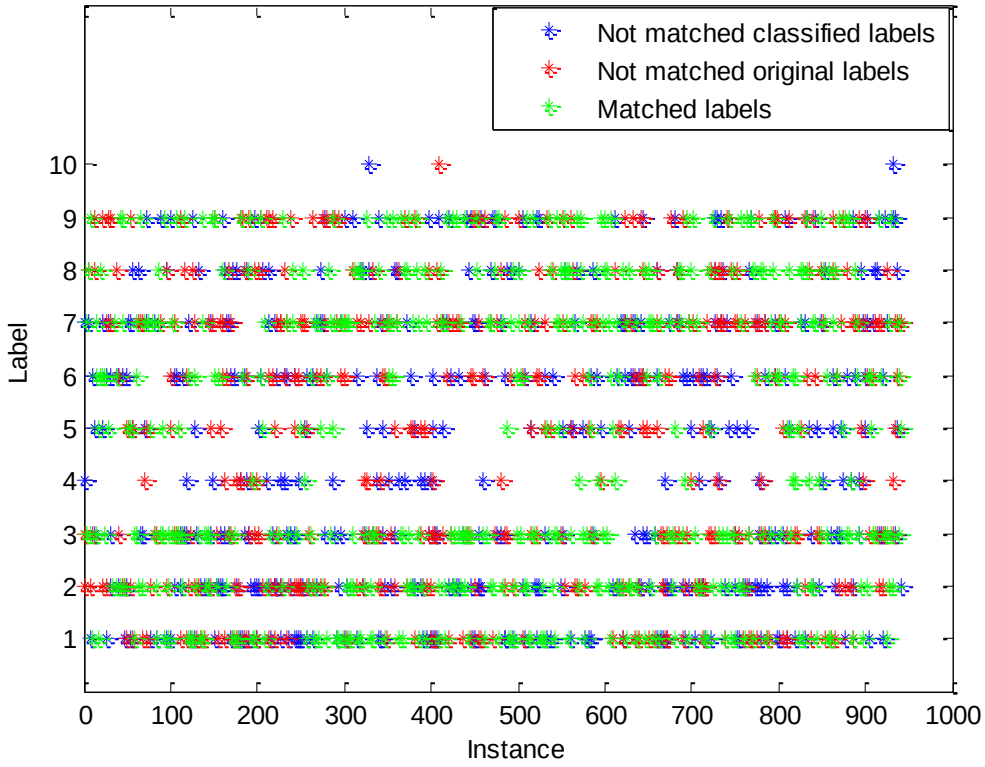


FIGURE 12 Results of the SVM classification for the dataset D2_reduced

Figures 13, 14 below represent amount of correctly predicted and misclassified labels in the accumulative bar format. Figure 13 presents absolute number of labels and Figure 14 presents their percentage. The first bar in each group corresponds to correctly predicted labels. The second bar is related to misclassified data. It represents total amount (or percentage) of misclassified data as well as the number (or percentage) grouped by labels. Each color stripe corresponds to a particular label.

These figures show that labels 1, 3, 7, 8, 9 have been predicted with accuracy higher that 50%, labels 2, 4, 5, 6 have been predicted with accuracy a little less than 50% and only data from label class 10 has not been correctly classified. Overall, this method shows quite good results.

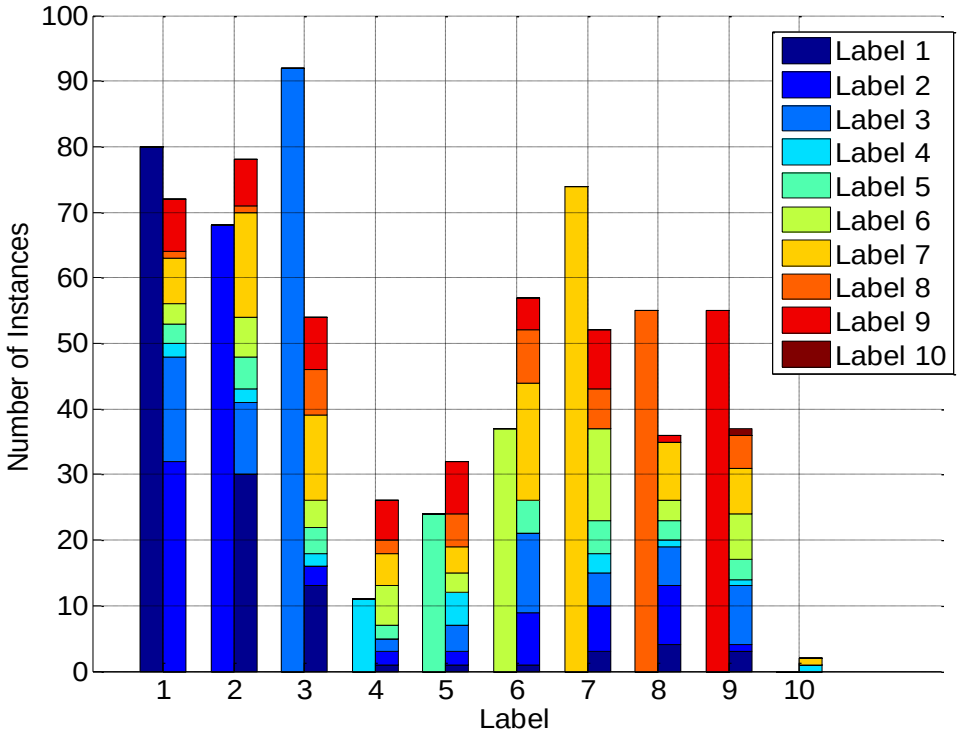


FIGURE 13 Number of correctly and incorrectly SVM classified data for the data set D2_reduced, groupped by labels

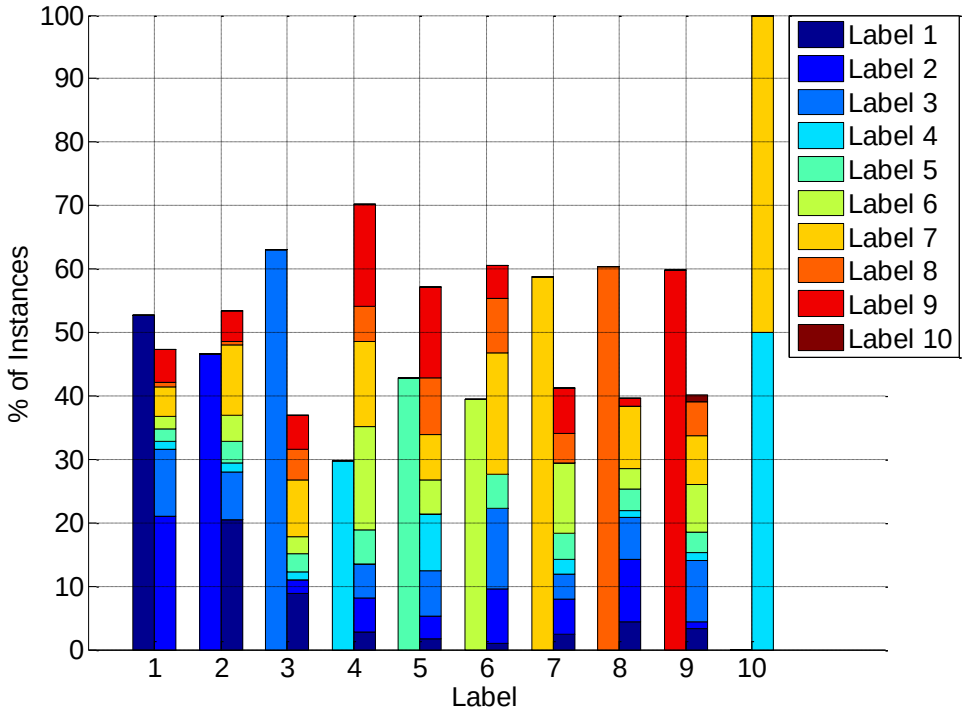


FIGURE 14 Percentage of correctly and incorrectly SVM classified data for the data set D2_reduced, groupped by labels

5.2.6 Supervised SOM

The data set D2 containing 24 features not related to applications has been chosen. The data set D2_reduced described in the section 5.2.1 has been used in the training and validation phases.

Validation technique used during implementation of the Supervised SOM is a mixture of holdout and 10-fold cross validation procedures. In a similar way to as described in the section 5.2.5, the data set D2_reduced first has been divided into training set (70%) and test set (30%). Then 10-fold cross validation has been performed on the training set in order to select parameters more precisely.

SOM toolbox for MATLAB (Vesanto et al., 2000) has been utilized for training, validation and estimation. The method has two most important parameters: preferred number of map units and map grid size. Default number of map units, or neurons, used in the SOM toolbox, is calculated by the formula:

$$Map\ units = 5 \times \sqrt{Number\ of\ instances}\,.$$

A number of experiments have been performed with different values for the grid map size from the interval $[5, 50]$. Figure 15 displays prediction accuracy corresponding to different values of the grid map size parameter. The experiments have shown that the best value for the grid map size is the tion (23,13).

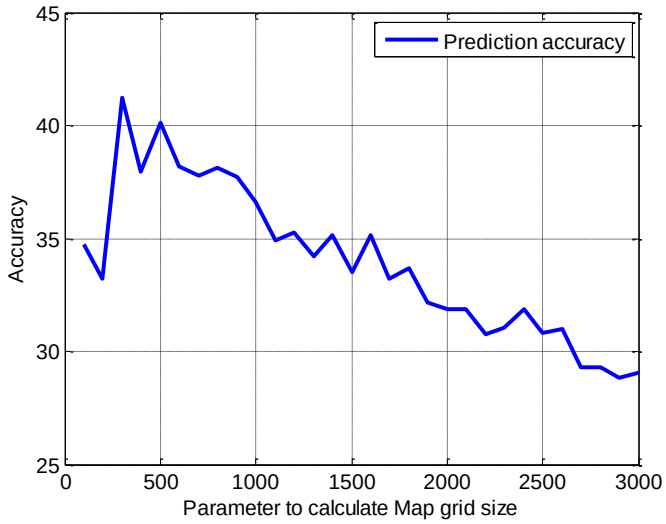


FIGURE 15 Results of the Map Grid Size parameter validation for the SOM

The final model has been trained on the whole training set using the selected parameters and has been tested on the test set. The classification accuracy of the obtained model can be seen in figures below.

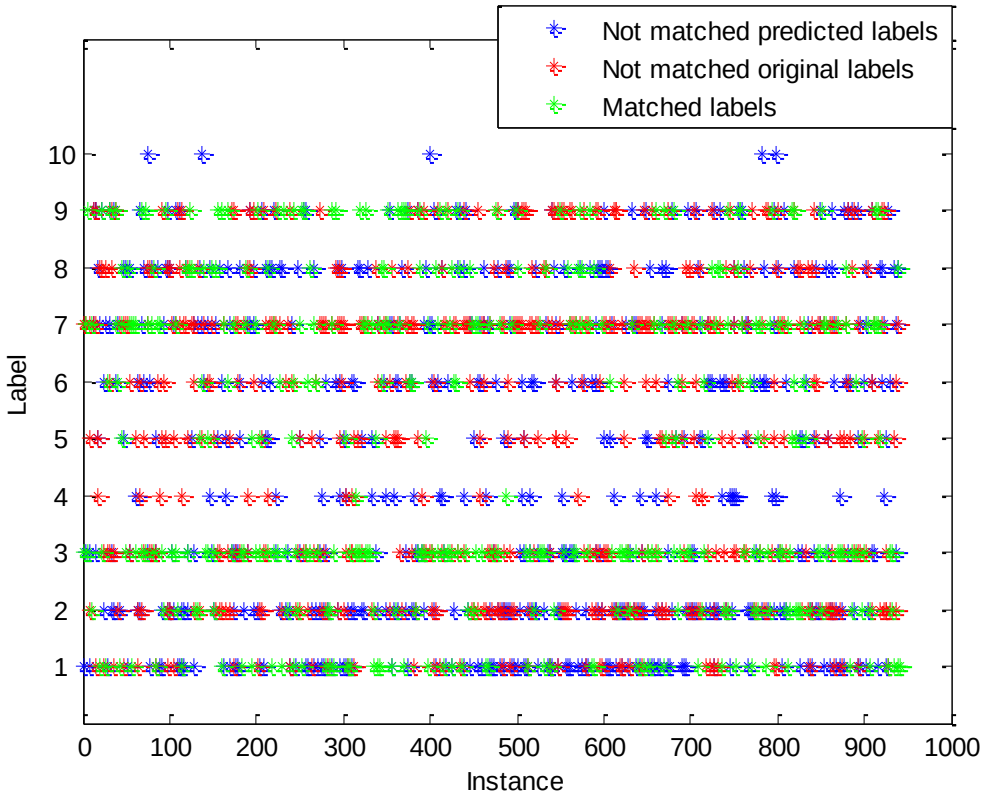


FIGURE 16 Results of the Supervised SOM classification for the dataset D2_reduced

Similarly as for SVM three visual representations of classification accuracy have been built. Figure 16 represents distinct labels separated by color into correctly classified and misclassified. Figures 17, 18 present amount and percentage of correctly and incorrectly predicted labels. Each label can be distinguished by color.

The method worked sufficiently well for almost all label classes, except for the class 10, which might be due to a relatively small number of instances of this class in the data set.

According to Figure 16, there are correctly classified data for every class, but 10^{th}. However, accuracy for the class 4 is less comparing to the others. From Figures 17, 18 it can be found out that only classes 3, 7 and 9 have prediction accuracy higher than 50%. Labels 1, 5, 8 have been predicted with accuracy around 35%, and prediction accuracy for the labels 2, 4, 6 is less than 30%. Data from the label class 10 has not been correctly classified at all. Overall accuracy of SOM on the test set is around 33%, which is lower than for SVM (around 45%).

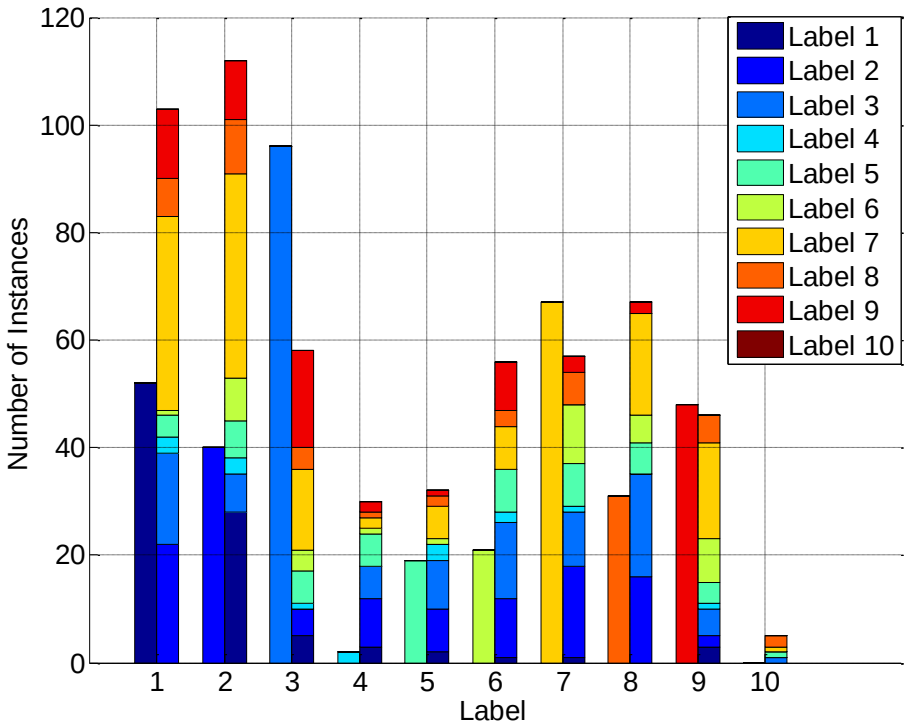


FIGURE 17 Number of correctly and incorrectly Supervised SOM classified data for the data set D2_reduced, groupped by labels

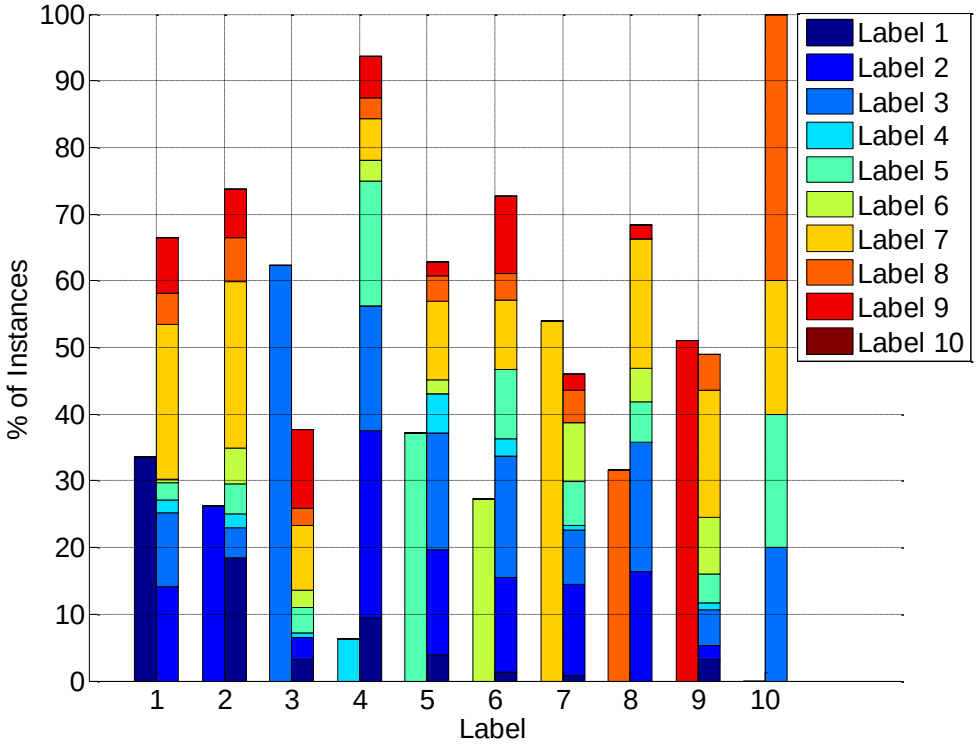


FIGURE 18 Percentage of correctly and incorrectly Supervised SOM classified data for the data set D2_reduced, groupped by labels

5.3 Analysis of the results

This section provides evaluation and comparison of the solutions developed in this research and compares the results obtained.

In the previous subsections the methods have been applied to the data set containing significant amount of records (32903). But along with a data set, an evaluation file, containing a list of place labels, has been provided as input for the current research. The structure of this evaluation file has been described in the section 4.2.2. The evaluation file is of size 336, which means number of all unique pairs "user id – place id" in the data set.

To perform solutions evaluation based on the provided evaluation file, it is needed to convert the prediction results, obtained through classification of the whole data set to the same format and size, as the evaluation file has.

For this purpose an instance-wise "voting" scheme has been introduced. According to this scheme, a label, which had been predicted the maximal number of times during classification of the whole data set, for a pair "user id – place id", has been chosen as place label for this pair. Thus, the final labels to be evaluated via the evaluation file, have first been predicted on the whole data set, and then went through the instance-wise "voting" procedure. In the end a file with predicted labels, where each combination of "user id – place id - label" is presented only once has been generated for further evaluation of the results.

There is a set of figures below representing evaluation results for three solutions, provided in the current research. Figures 19-21 represent distinct labels separated by color into not matched classified, not matched original and matched labels. The term "not matched classified labels" is related to the correct labels, obtained from the evaluation file. The term "not matched original labels" is related to incorrectly predicted values, compared to those from the evaluation file. There is a one-to-one relationship between not matched classified labels and not matched original labels. Matched labels are marked green and mean that these labels have been predicted correctly.

In Figure 19, labels from the evaluation file are compared with the ones obtained through classification via the Naïve Bayes, combined with the instance-wise "voting". Similarly to the Naïve Bayes classification without "voting", the correctly predicted labels belong to the classes 1, 3 and 9. However, class 2 has now disappeared from the set of matched labels. This can signify that the accuracy of predicting class 2 for a proper pair "user id – place id" is less than accuracy of predicting some other class.

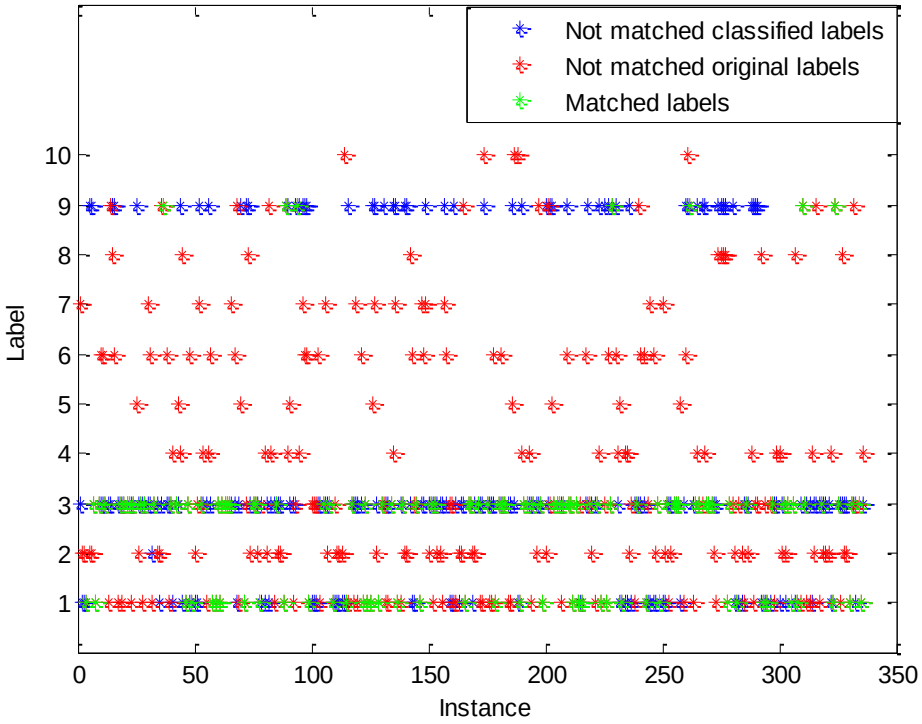


FIGURE 19 Comparison of the evaluation file and Naïve Bayes plus instance-wise "voting" classification

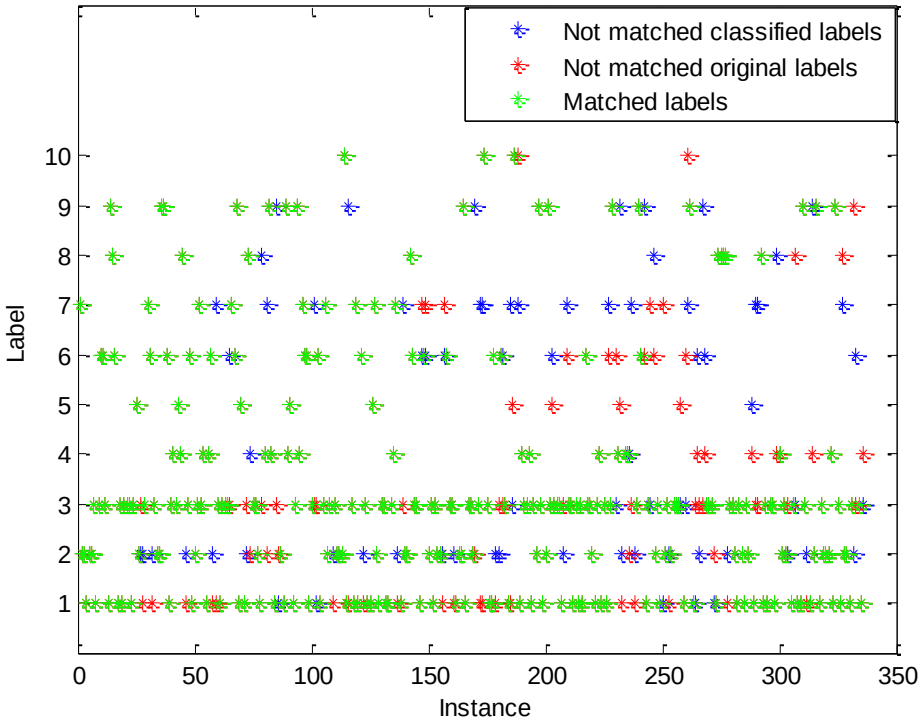


FIGURE 20 Comparison of the evaluation file and SVM plus instance-wise "voting" classification

In Figure 20, labels from the evaluation file are compared with the ones, obtained through classification via the SVM, combined with the instance-wise "voting". The results are similar to the SVM classification without "voting", which is all classes are predicted quite good. Moreover, instance-wise "voting" has increased prediction accuracy for the class 10, and thus overall prediction accuracy has increased.

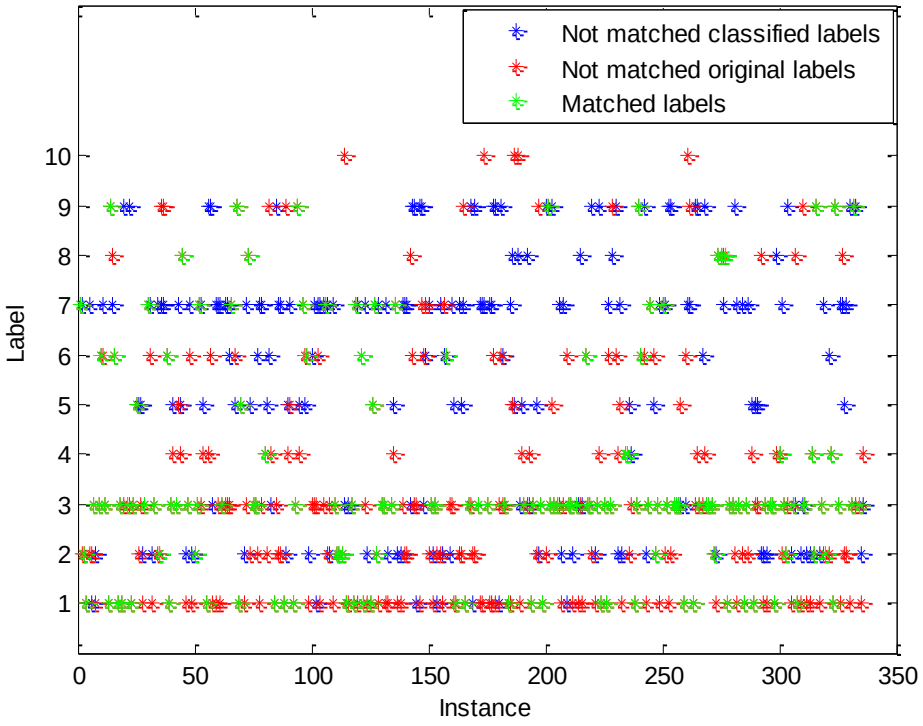


FIGURE 21 Comparison of the evaluation file and Supervised SOM plus instance-wise "voting" classification

In Figure 21, labels from the evaluation file are compared with the ones, obtained through classification via the Supervised SOM, combined with the instance-wise "voting". The results are similar to the Supervised SOM classification without "voting", which is all classes excluding the class 10 are predicted with average accuracy around 30%.

Table 3 presents performance evaluation results for all three solutions, presented in this research. Precision, Recall and F1 score have been calculated for separate classes, also total F1 score has been calculate as average F1 score for all classes, according to the formula:

$$F_1 score\ total = \frac{1}{10}\sum_{i=1}^{10} F_1 score\ (label == i)\ ,$$

where $F_1 score\ (label == i)$ is F1 score value for the i^{th} class.

TABLE 3 Performance evaluation results of (a) Naïve Bayes plus instance-wise "voting", (b) Supervised SOM plus instance-wise "voting", (c) SVM plus instance-wise "voting"

Label	Precision	Recall	F1 score
1	0,44	0,48	0,46
2	0	0	0
3	0,42	0,75	0,54
4	0	0	0
5	0	0	0
6	0	0	0
7	0	0	0
8	0	0	0
9	0,11	0,41	0,18
10	0	0	0
Total:			**0.12**

(a)

Label	Precision	Recall	F1 score
1	0,82	0,37	0,51
2	0,27	0,26	0,26
3	0,81	0,65	0,72
4	0,86	0,26	0,4
5	0,12	0,33	0,17
6	0,47	0,32	0,38
7	0,15	0,79	0,25
8	0,45	0,45	0,45
9	0,22	0,47	0,3
10	0	0	0
Total:			**0.34**

(b)

Label	Precision	Recall	F1 score
1	0,93	0,76	0,84
2	0,65	0,87	0,74
3	0,92	0,81	0,86
4	0,89	0,74	0,81
5	0,83	0,56	0,67
6	0,68	0,76	0,72
7	0,38	0,64	0,47
8	0,75	0,82	0,78
9	0,7	0,94	0,8
10	1	0,6	0,75
Total:			**0.74**

(c)

Thus, the best solution is the one with the highest total F1 score. The highest total F1 score value 0.74 corresponds to the SVM plus instance-wise "voting" classification. Other solutions have shown worse results with total F1 score of 0.34 for the Supervised SOM plus instance-wise "voting" classification and total F1 score of only 0.12 for the Naïve Bayes plus instance-wise "voting" classification.

According to Table 3 the winning classification scheme has predicted all classes with sufficiently high accuracy, whereas other schemes have rather dissimilar results depending on the class.

Thus, the Naïve Bayes plus instance-wise "voting" classification has ensured nonzero F1 score only for the classes 1, 3, 9. This can be due to the several reasons: (1) there is no distinctive set of applications running for particular clas-

ses, (2) the features correlate, and (3) huge domination of the classes with labels 1, and 3.

Similarly, the Supervised SOM plus instance-wise "voting" classification has ensured F1 score greater than 0.5 only for the classes 1 and 3. Such small accuracy can be explained by several reasons: (1) fuzzy clusters boundary among classes of the data, and (2) huge domination of the classes with labels 1, and 3.

6 CONCLUSIONS AND FUTURE WORK

Knowledge of user activities is very important for development of personalized applications. Predicting location of mobile users enables development of high quality location-based services and applications, and assists in improving resource reservation in wireless networks.

Main goal of the research was to develop a solution for predicting semantic meaning of a set of places extracted from mobile phone data, collected from 80 users during quite long period of time. The most important features related to users' activities have been extracted from the data set. Therefore, 410 features have been extracted, 386 out of them are binary features related to applications. Based on the extracted features several training and test sets have been formed. Three approaches based on the Naïve Bayes, Supervised Self-Organizing Maps and Support Vector Machine methods have been implemented and compared. The results obtained by the suggested solutions have been evaluated using an evaluation file provided along with a data set as input to this research work. The highest F1 score value of 0.74 for the predicted labels of the most significant places has shown the approach based on the Support Vector Machine. Other methods have shown less positive accuracy values. Thus, in terms of F1 measure, the approach based on Supervised Self-Organizing Maps has got 0.34, and the solution based on the Naïve Bayes method has got only 0.12.

The research can be developed further. Although a huge amount of features have been extracted from the raw data, still more features can be obtained. For example, instead of only aggregate avdelta parameter from the accelerometer data, a full range of measurements related to all spatial dimensions and time can be considered individually.

Also different combinations of methods can be tried including smart method-wise voting in the postprocessing phase. It is worth trying to improve

the instance-wise voting scheme used in the postprocessing phase of the current research through smart instance-wise voting schemes, where different weights could be assigned for classes.

Apart from applying classical data mining classification techniques considered in this Master's thesis, the results, which can be obtained by analysis of movement patterns, can be integrated in the proposed solutions. Such movement patterns can be constructed based on GPS measurement or base station identification numbers combined with visiting time stamps. The transition models can be built on the basis of Bayesian networks. Such combinations can significantly increase classification accuracy of the solutions.

REFERENCES

Abe, S., 2010. Support Vector Machines for Pattern Classification, 2nd ed. ed. Springer.

Agrawal, R., Srikant, R., 1995. Mining Sequential Patterns, in: Proceedings of the Eleventh International Conference on Data Engineering, ICDE '95. IEEE Computer Society, Washington, DC, USA, pp. 3–14.

Anagnostopoulos, T., Anagnostopoulos, C., Hadjiefthymiades, S., 2011. An adaptive location prediction model based on fuzzy control. Comput. Commun. 34, 816–834.

Asahara, A., Maruyama, K., Sato, A., Seto, K., 2011. Pedestrian-movement prediction based on mixed Markov-chain model, in: Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS '11. ACM, New York, NY, USA, pp. 25–33.

Ashbrook, D., Starner, T., 2003. Using GPS to learn significant locations and predict movement across multiple users. Personal Ubiquitous Comput. 7, 275–286.

Bayes, T., Price, R., Molina, E.C., Canton, J., Deming, W.E., 1940. Facsimiles of two papers by Bayes: I. An essay toward solving a problem in the doctrine of chances, with Richard Price's forward and discussion; Phil. Trans. Royal Soc., pp.370-418, 1763. With a commentary by Edward C. Molina. II. A letter on asymptotic series from Bayes to John Canton; pp.269-271 of the same volume. With a commentary by W. Edwards Deming. Hafner Pub. Co.

Bayir, M.A., Demirbas, M., Eagle, N., 2010. Mobility profiler: A framework for discovering mobility profiles of cell phone users. Pervasive and Mobile Computing 6, 435–454.

Bernardo, J. & Smith, A.F.M., 2007. Bayesian Theory. John Wiley & Sons, Incorporated

Bhattacharya, A., Das, S.K., 1999. LeZi-update: an information-theoretic approach to track mobile users in PCS networks, in: Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking, MobiCom '99. ACM, New York, NY, USA, pp. 1–12.

Buessler, J.-L., Urban, J.-P., Gresser, J., 2002. Additive Composition of Supervised Self-Organizing Maps. Neural Process. Lett. 15, 9–20.

Byun, H., Lee, S.-W., 2002. Applications of Support Vector Machines for Pattern Recognition: A Survey, in: Proceedings of the First International Workshop on Pattern Recognition with Support Vector Machines, SVM '02. Springer-Verlag, London, UK, UK, pp. 213–236.

Chang, C.-C., Lin, C.-J., 2011. LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27.

Cristianini, N., Shawe-Taylor, J., 2000. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press.

David, G., 2012. Anomaly Detection Theory, Algorithms and Applications. Lecture notes.

Davis, J.A., Fagg, A.H., Levine, B.N., 2001. Wearable computers as packet transport mechanisms in highly-partitioned ad-hoc networks, in: Fifth International Symposium on Wearable Computers, 2001. Proceedings. Presented at the Fifth International Symposium on Wearable Computers, 2001. Proceedings, pp. 141 –148.

Domingos, P., Pazzani, M., Provan, G., 1997. On the Optimality of the Simple Bayesian Classifier under Zero-One Loss, in: Machine Learning. pp. 103–130.

Duda, R.O., Hart, P.E., Stork, D.G., 2000. Pattern Classification, 2nd ed. Wiley-Interscience.

Gambs, S., Killijian, M.-O., del Prado Cortez, M.N., 2010. Show me how you move and I will tell you who you are, in: Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS, SPRINGL '10. ACM, New York, NY, USA, pp. 34–41.

Gambs, S., Killijian, M.-O., del Prado Cortez, M.N., 2012. Next place prediction using mobility Markov chains, in: Proceedings of the First Workshop on Measurement, Privacy, and Mobility, MPM '12. ACM, New York, NY, USA, pp. 3:1–3:6.

Hagenbuchner, M., Tsoi, A., 2004. A supervised self-organizing map for structures. Faculty of Informatics - Papers.

Hastie, T., Tibshirani, R., Friedman, J., 2009. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition, 2nd ed. 2009. Corr. 3rd printing 5th Printing. ed. Springer.

Hoh, B., Gruteser, M., Herring, R., Ban, J., Work, D., Herrera, J.-C., Bayen, A.M., Annavaram, M., Jacobson, Q., 2008. Virtual trip lines for distributed privacy-preserving traffic monitoring, in: Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services, MobiSys '08. ACM, New York, NY, USA, pp. 15–28.

Hudson, J.M., Christensen, J., Kellogg, W.A., Erickson, T., 2002. "I'd be overwhelmed, but it's just one more thing to do": availability and interruption in research management, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Changing Our World, Changing Ourselves, CHI '02. ACM, New York, NY, USA, pp. 97–104.

Hull, B., Bychkovsky, V., Zhang, Y., Chen, K., Goraczko, M., Miu, A., Shih, E., Balakrishnan, H., Madden, S., 2006. CarTel: a distributed mobile sensor computing system, in: Proceedings of the 4th International Conference on Embedded Networked Sensor Systems, SenSys '06. ACM, New York, NY, USA, pp. 125–138.

Isaacman, S., Becker, R., Cáceres, R., Kobourov, S., Martonosi, M., Rowland, J., Varshavsky, A., 2011. Identifying important places in people's lives from cellular network data, in: Proceedings of the 9th International Conference on Pervasive Computing, Pervasive'11. Springer-Verlag, Berlin, Heidelberg, pp. 133–151.

Jeffreys, H., 1998. Theory of Probability. Oxford University Press, USA.

Kang, J.H., Welbourne, W., Stewart, B., Borriello, G., 2005. Extracting places from traces of locations. SIGMOBILE Mob. Comput. Commun. Rev. 9, 58–68.

Kantola, J., Perttunen, M., Leppanen, T., Collin, J., Riekki, J., 2010. Context Awareness for GPS-Enabled Phones, in: Proceedings of the 2010 International Technical Meeting of The Institute of Navigation, pp. 117-124.

Kim, B., Ha, J.-Y., Lee, S., Kang, S., Lee, Y., Rhee, Y., Nachman, L., Song, J., 2011. AdNext: a visit-pattern-aware mobile advertising system for urban commercial complexes, in: Proceedings of the 12th Workshop on Mobile Computing Systems and Applications, HotMobile '11. ACM, New York, NY, USA, pp. 7–12.

Knerr, S., Personnaz, L., Dreyfus, G., 1990. Single-Layer Learning Revisited: A Stepwise Procedure for Building and Training a Neural Network, in: Fo-

gelman Soulié, F., Hérault, J. (Eds.), Neurocomputing: Algorithms, Architectures and Applications, NATO ASI Series. Springer-Verlag, pp. 41–50.

Kohonen, T., 2001. Self-Organizing Maps. Springer.

Kortuem, G., Schneider, J., Suruda, J., Fickas, S., Segall, Z., 1999. When Cyborgs Meet: Building Communities of Cooperating Wearable Agents, in: Proceedings of the 3rd IEEE International Symposium on Wearable Computers, ISWC '99. IEEE Computer Society, Washington, DC, USA, p. 124.

Laurila, J.K., Gatica-Perez, D., Aad, I., Blom, J., Bornet, O., Do, T.-M.-T., Dousse, O., Eberle, J., Miettinen, M., 2012. The Mobile Data Challenge: Big Data for Mobile Computing Research, in: Proc. Mobile Data Challenge by Nokia Workshop, in conjunction with Int. Conf.. on Pervasive Computing, Newcastle.

Liao, L., Fox, D., Kautz, H., 2005. Location-Based Activity Recognition using Relational Markov Networks, in: In: Proceedings of the Nineteenth International Conference on Artificial Intelligence, IJCAI'05.

Liu, G.Y., Maguire, G.Q., J., 1995. Efficient mobility management support for wireless data services, in: Vehicular Technology Conference, 1995 IEEE 45th. Presented at the Vehicular Technology Conference, 1995 IEEE 45th, pp. 902 –906 vol.2.

Mager, J., Paasche, U., Sick, B., 2008. Forecasting financial time series with support vector machines based on dynamic kernels, in: IEEE Conference on Soft Computing in Industrial Applications, 2008. SMCia '08. Presented at the IEEE Conference on Soft Computing in Industrial Applications, 2008. SMCia '08, pp. 252 –257.

Maimon, O.Z., Rokach, L., 2005. Data Mining And Knowledge Discovery Handbook. Springer.

Marmasse, N., Schmandt, C., 2000. Location-Aware Information Delivery with ComMotion, in: Proceedings of the 2nd International Symposium on Handheld and Ubiquitous Computing, HUC '00. Springer-Verlag, London, UK, UK, pp. 157–171.

Mohan, P., Padmanabhan, V.N., Ramjee, R., 2008. Nericell: rich monitoring of road and traffic conditions using mobile smartphones, in: Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems, SenSys '08. ACM, New York, NY, USA, pp. 323–336.

NG, A., 2012. Machine Learning. Course Materials.

Papliatseyeu, A., Mayora, O., n.d. Mobile Habits: Inferring and Predicting User Activities with a Location-Aware Smartphone, in: Corchado, J.M., Tapia, D.I., Bravo, J. (Eds.), 3rd Symposium of Ubiquitous Computing and Ambient Intelligence 2008. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 343–352.

Patterson, D.J., Liao, L., Fox, D., Kautz, H., 2003. Inferring High-Level Behavior from Low-Level Sensors. pp. 73–89.

Pearson, K., 1901. On lines and planes of closest fit to systems of points in space. Philosophical Magazine 2, 559–572.

Rijsbergen, C.J.V., 1979. Information Retrieval, 2nd ed. Butterworth-Heinemann, Newton, MA, USA.

Roth, J., Unger, C., 2001. Using Handheld Devices in Synchronous Collaborative Scenarios. Personal Ubiquitous Comput. 5, 243–252.

Sparacino, F., 2002. The Museum Wearable: real-time sensor-driven understanding of visitors' interests for personalized visually-augmented museum experiences, in: In: Proceedings of Museums and the Web MW2002. pp. 17–20.

Steinwart, I., Christmann, A., 2008. Support Vector Machines. Springer.

Tay F.E.H., Cao L.J., 2002. Modified support vector machines in financial time series forecasting. Neurocomputing 48, 847–861.

Terry, M., Mynatt, E.D., Ryall, K., Leigh, D., 2002. Social net: using patterns of physical proximity over time to infer shared interests, in: CHI '02 Extended Abstracts on Human Factors in Computing Systems, CHI EA '02. ACM, New York, NY, USA, pp. 816–817.

Vapnik, V.N., 1995. The nature of statistical learning theory. Springer-Verlag New York, Inc., New York, NY, USA.

Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J., 2000. Self-Organizing Map in Matlab: the SOM Toolbox, in: In Proceedings of the Matlab DSP Conference. pp. 35–40.

Witten, I.H., Frank, E., 2005. Data Mining: Practical Machine Learning Tools and Techniques, Second Edition. Morgan Kaufmann.

Wolf, J., Guensler, R., Bachman, W., n.d. Elimination of the Travel Diary: An Experiment to Derive Trip Purpose From {GPS} Travel Data. Notes from

Transportation Research Board, 80th annual meeting, January 7{--}11, 2001, Washington, D.C.

Yoon, J., Noble, B., Liu, M., 2007. Surface street traffic estimation, in: Proceedings of the 5th International Conference on Mobile Systems, Applications and Services, MobiSys '07. ACM, New York, NY, USA, pp. 220–232.

Zhang, K., Li, H., Torkkola, K., Gardner, M., 2007. Adaptive learning of semantic locations and routes, in: Proceedings of the 1st International Conference on Autonomic Computing and Communication Systems, Autonomics '07. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, pp. 3:1–3:10.

Zhou, C., Frankowski, D., Ludford, P., Shekhar, S., Terveen, L., 2004. Discovering personal gazetteers: an interactive clustering approach, in: Proceedings of the 12th Annual ACM International Workshop on Geographic Information Systems, GIS '04. ACM, New York, NY, USA, pp. 266–273.

Zhuang, Z., Kim, K.-H., Singh, J.P., 2010. Improving energy efficiency of location sensing on smartphones, in: Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, MobiSys '10. ACM, New York, NY, USA, pp. 315–330.