Identifying Hidden Contexts in Classification

Indrė Žliobaitė

Eindhoven University of Technology, Eindhoven, the Netherlands
Smart Technology Research Center, Bournemouth University, Poole, UK
zliobaite@gmail.com

Abstract. In this study we investigate how to identify hidden contexts from the data in classification tasks. Contexts are artifacts in the data, which do not predict the class label directly. For instance, in speech recognition task speakers might have different accents, which do not directly discriminate between the spoken words. Identifying hidden contexts is considered as data preprocessing task, which can help to build more accurate classifiers, tailored for particular contexts and give an insight into the data structure. We present three techniques to identify hidden contexts, which hide class label information from the input data and partition it using clustering techniques. We form a collection of performance measures to ensure that the resulting contexts are valid. We evaluate the performance of the proposed techniques on thirty real datasets. We present a case study illustrating how the identified contexts can be used to build specialized more accurate classifiers.

1 Introduction

In classification tasks some variables directly predict the class label, others can describe context. Contexts are artifacts in the data, which do not directly predict the class label, like accent in speech recognition. Taking contexts into the learning process can help to build more specialized and accurate classifiers [2], solve sample selection bias [12], concept drift [17] problems.

Context may not necessarily be present in a form of a single variable in the feature space. To recover *hidden contexts* the input data can be clustered [5,9,15]. The problem is, that clustering can capture some class label information, which would shade away the contexts. Consider a diagnostics task, where patient tests are taken by two pieces of equipment, which are calibrated differently. If we cluster patient data, the resulting clusters might correspond to 'healthy' and 'sick' (which are the classes) or 'sample taken by equipment A' and 'sample taken by equipment B' (which is a context), but likely a mix of both. Thus, to capture context specific information, we intend to force independence between contexts and class labels. In addition, capturing noise is undesired, therefore the resulting contexts need to be non-random and stable.

Context identification has predictive and descriptive goals. Grouping the data provides an opportunity to achieve more accurate classification employing context handling strategies, as well as better understand the phenomenon behind

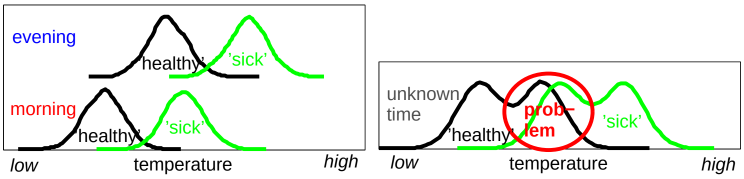

Fig. 1. An example of two contexts.

the data. Context identification can be considered as a preprocessing step in supervised learning, like feature selection, instance selection, or recovering missing values. We aim for a filter approach, where contexts are generic, not tied with a particular handling strategy.

In this study we propose three techniques for identifying hidden contexts, which force independence between the contexts and class labels. The objective is to output an explicit grouping of the data. We require the grouping to ignore the class label information. Thus, we aim to hide class discriminatory information before partitioning. These techniques can be used in different context handling strategies or for forming new ones.

We analyze the performance of the proposed techniques on thirty real datasets. We also present a case study, which illustrates one example strategy for handling the identified contexts in classification.

The paper is organized as follows. Section 2 defines context. Section 3 discusses related work. In Section 4 we propose three techniques for identifying hidden contexts. Sections 5 and 6 present experimental evaluation. Section 7 concludes the study.

2 Problem set-up

Consider a classification problem in p-dimensional space. Given a set of instances with labels ($\mathbf{X} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}$) the task is to produce a classifier $h : \mathcal{X} \rightarrow \mathcal{Y}$. In this study we define context as a secondary label z of an instance X, which is independent from the class label y, but can explain the class label better when used together with the predictive features. That is, $p(y|z) = p(y)$, but $p(y|X^*, z) \neq p(y|X^*)$, where $X^* \subseteq X$. Context might be expressed as a variable in the feature space (known context) or as a latent variable (unknown context).

Consider as a toy example a task, where a patient is diagnosed 'sick' or 'healthy' based on the body temperature. It is known that in the evening people tend to have higher temperature *independently* of being sick or healthy. If we know the context, i.e. whether the temperature was measured in the morning or in the evening, diagnostic task is easy, as illustrated in Figure 1. However, if the time is unknown, then diagnosing becomes problematic. The time itself is independent from the class label, stand alone it does not diagnose.

Some more examples of context include accent in speech recognition, light in image recognition, seasonality in sales prediction, weekday in electricity load or bus travel time prediction, industry crisis in credit scoring.

The context label may be *directly observable* or *hidden*, depending on the application. Hidden context variable z is not explicitly present as a feature x in the feature space $\mathcal{X}$, but information about it is assumed to be captured in the feature space $\mathcal{X}$, i.e. $z = f(X)$, $X \in \mathcal{X}$. An example of directly observable context is time. Customer segments in marketing tasks or bankruptcy prediction represent hidden context. Different segments might have different behavior.

Evaluation of the identified contexts is not straightforward. Different context handling strategies can lead to different gains or losses in the classification accuracies, which are not necessarily due to good or bad context identification. We require the resulting contexts to be independent from the class labels, valid (not random grouping of the data) and stable on random subsamples of the same data. The criteria to measure these aspects are formulated in Section 5.

3 Positioning within related work

Context-awareness is widely used in ubiquitous and pervasive computing to characterize the environmental variables [14]. In machine learning the term usually characterizes the features that do not determine or influence the class of an object directly [2, 15]. A wide body of literature on concept drift considers only time context [7, 17]. Typically contexts assumed to be known. Mixture models (see [4]) can be considered as an approach to identify hidden contexts.

Our context identification techniques are novel as they force independency from the class labels. A need for such approaches was mentioned before in a light of context handling strategies [16] and multiple classifier systems [3]. Turney [16] formulated the problem of recovering implicit context information and proposed two techniques: input data clustering and time sequence (which we leave out of the scope assuming that the chronological order is unknown). Turney expressed a concern that clustering might capture class label information and indicated a need for further research, our work can be seen as a follow up.

A recent work by Dara et al [3] explores the relation between the characteristics of data partitions and final model accuracy in multiple classifier systems. The work experimentally confirms the benefits of partitions which are not correlated with the class labels. These results support the motivation of our work.

As a result of their analysis Dara et al [3] propose a semi-randomized partitioning strategy to cluster and then swap some instances across the clusters, which can be seen as a mixture of clustering and boosting. We excluded this strategy from our investigations after preliminary experiments, since even though it pushes towards independence in class labels within the partitions (which is our objective as well), due to randomization the procedure of assigning an unseen instance to one of the partitions can no longer be deterministic.

Extracting hidden context is related to the context handling strategies [16]. The strategies are not limited to building a separate classifier or a combination

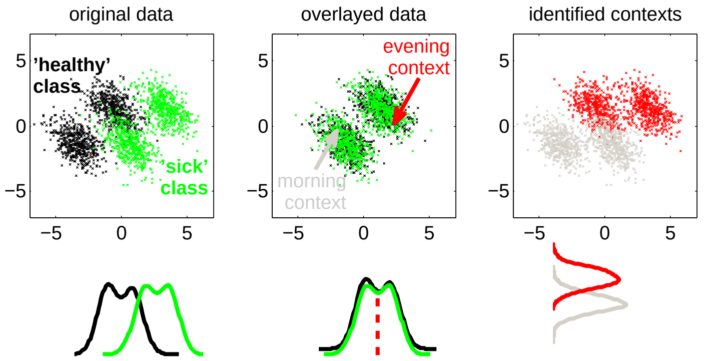

Fig. 2. An illustration of *Overlay* technique.

for each context. Contextual information can be also used to adjust the input data, model parameters or model outputs. Analysis of the performance of different handling strategies is out of the cope of this study. The identified and validated contexts can be used as building blocks to handling strategies.

4 Three techniques for identifying hidden contexts

In this section we present techniques for identifying hidden contexts. Given a dataset, the task is to allocate the instances into k groups, so that the data within the groups is related, but the groups are not related to the class labels.

Context identification techniques require two mechanisms: (1) how to group the training data $\mathbf{X}$ into k contexts and (2) how to assign an unseen instance $X \notin \mathbf{X}$ to one of the contexts.

Clustering (CLU) of the input data is the baseline technique to identify hidden contexts, when building local models [5, 9–11]. The procedure is summarized in Figure 4. Clustering captures closeness of the data instances in the feature space. For classification tasks the feature space is typically formed with an intention to predict the class labels. If class membership information is strongly present in the data, clustering is likely to capture it as well.

To overcome this issue we propose **Overlay (OVE)** technique. To hide the label information we move the classes on top of each other, as illustrated in Figure 2, by normalizing each class to zero mean. The technique assumes that class discriminatory information lies in the class means. We cluster the overlayed data to extract contexts. Unfortunately, for the incoming new data we cannot do overlay, because the labels are unknown. We solve this by introducing a supervised context learning. Given the instances $\mathbf{X}$ we treat the obtained contexts $\mathbf{z}$ as labels and learn a classifier $z = \mathcal{K}_{OVE}(X)$. We use the diagonal linear discriminant [4] as a classifier $\mathcal{K}_{OVE}$. The procedure is summarized in Figure 4.

Overlay technique is based on the assumption that the class distributions are symmetric across different contexts, which often might not be true. We generalize

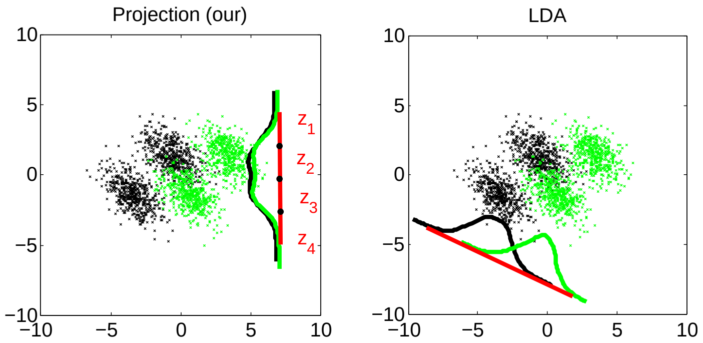

Fig. 3. An illustration of *Projection* technique.

Overlay by introducing **Projection (PRO)** technique, which rotates the data to hide the class label information. The idea is opposite to Linear Discriminant Analysis (LDA) [4]. The goal is to find a transformation that minimizes between-class variance and maximizes within-class variance, see Figure 3.

We seek to find a transformation $\mathbf{w}$ to obtain a projection $\breve{\mathbf{x}} = \mathbf{w}'\mathbf{X}$. Within-class c_i covariance is $s_i^2 = \sum_{y_j = c_i} (\mathbf{X}_j - \mu_i)(\mathbf{X}_j - \mu_i)'$, where μ_i is the class mean. The total within class covariance is $S_s := s_1^2 + s_2^2 + \ldots + s_c^2$. Between-class covariance is $S_b := \frac{1}{c}\sum_{i=1}^{c}(\mu_i - \mu)(\mu_i - \mu)'$, where μ is the mean of all the data.

In LDA Fisher criterion $J(w) = \frac{\mathbf{w}' S_b \mathbf{w}}{\mathbf{w}' S_s \mathbf{w}}$ is maximized. We minimize it. The problem transforms into eigenvalue decomposition $S_s^{-1} S_b \mathbf{w} = \lambda \mathbf{w}$. We choose the eigenvector $\mathbf{w}$ corresponding to the smallest eigenvalue $\min \lambda$. To determine contexts $\mathbf{z}$, we transform the training data into 1D space $\breve{\mathbf{x}} = \mathbf{w}'\mathbf{X}$ and simply split the range of values into k equal intervals (like slicing a loaf of bread). An unseen instance X is transformed into 1D $\breve{x} = \mathbf{w}X$ and assigned a context, based on the interval, to which it falls into. The procedure is presented in Figure 4.

In addition to Overlay and Projection, we explore **Feature underselection (FUS)** technique, which discards the features, that are the most correlated with the class label, and clusters the remaining features. It is described in Figure 4.

All the presented techniques assume that k is given. In the case study (Section 6) we will show, how k can be determined using the stability criterion.

5 Experimental Evaluation

The goal of the experiments is to compare the introduced techniques in terms of the desired properties: not to capture the class labels, at the same time controlling, that the resulting partitions are valid and stable.

Clustering (CLU)
input: dataset $\mathbf{X}$; number of contexts k.
output: context labels $\mathbf{z}$; the rule for assigning unseen instances$z = \mathcal{K}_{CLU}(X)$.

1. Cluster the dataset to obtain contexts $\mathbf{z} = \mathit{clust}(\mathbf{X}, k)$, where $\mathit{clust}(., k)$ is any distance based clustering algorithm.
2. Fix the rule for unseen instances $\mathcal{K}_{CLU}(X) : z = \arg\min_{i=1..k} \mathit{dist}(X, C_i)$, where $C_1, \dots, C_k \in \mathcal{X}$ are the resulting cluster centers and $\mathit{dist}()$ is a distance function corresponding to the chosen clustering algorithm.

Overlay (OVE)
input: labeled dataset $(\mathbf{X}, \mathbf{y})$; number of contexts k.
output: context $\mathbf{z}$; rule for unseen instances $z = \mathcal{K}_{OVE}(X)$.

1. Split $\mathbf{X}$ into c groups: $X_j \in \mathbf{X}_i$ if $y_j = i, \forall X \in \mathbf{X}$, c is the number of classes.
2. Shift each class to *zero* mean: for $i = 1 \dots c$ $\hat{\mathbf{X}}_i = \mathbf{X}_i - \mu_i$, where μ_i is the mean of class c_i.
3. Overlay the classes $\hat{\mathbf{X}} = \{\hat{\mathbf{X}}_1 \cup \dots \cup \hat{\mathbf{X}}_c\}$.
4. Cluster $\hat{\mathbf{X}}$ to obtain the contexts $\mathbf{z} = \mathit{clust}(\hat{\mathbf{X}}, k)$.
5. Learn a classifier $z = \mathcal{K}_{OVE}(X)$ using $\mathbf{X}$ as input data and $\mathbf{z}$ as labels.

Projection (PRO)
input: labeled dataset $(\mathbf{X}, \mathbf{y})$; number of contexts k.
output: context $\mathbf{z}$; rule for unseen instances $z = \mathcal{K}_{PRO}(X)$.

1. Find a transformation vector $\mathbf{w}$, corresponding to the smallest eigenvalue $\min \lambda$ in $S_s^{-1} S_b \mathbf{w} = \lambda \mathbf{w}$, where S_s is within-class covariance and S_b is between-class covariance.
2. Transform into 1D space: $\breve{\mathbf{x}} = \mathbf{w}'\mathbf{X}$.
3. For $j = 1 \dots k$ find the intervals $r_j = \breve{x}_{min} + q(j-1)$, where $q = (\breve{x}_{max} - \breve{x}_{min})/k$.
4. Find context labels $z = j | \breve{x} \in r_j, \forall \breve{x} \in \breve{\mathbf{x}}$.
5. Fix the rule $\mathcal{K}_{PRO}(X) : z = j | \breve{x} \in r_j$, $\breve{x} = w'X$.

Feature underselection (FUS)
input: labeled dataset $(\mathbf{X}, \mathbf{y})$; a number of contexts k; number of features to select m.
output: context $\mathbf{z}$; rule for unseen instances $z = \mathcal{K}_{FUS}(X)$.

1. For $i = 1 \dots p$ find $\rho_i = \mathit{corr}(\mathbf{x}_i, \mathbf{y})$, where p is the dimensionality, $\mathbf{x}_i$ is the i^{th} dimension of the data.
2. Sort correlations: $|\rho_{i1}| \leq |\rho_{i2}| \leq \dots \leq |\rho_{ip}|$.
3. Pick m dimensions, the least correlated with class labels: $\tilde{\mathbf{X}} = (\mathbf{x}_{i1}\mathbf{x}_{i2} \dots \mathbf{x}_{im})'$.
4. Cluster $\tilde{\mathbf{X}}$ to obtain the contexts $\mathbf{z} = \mathit{clust}(\tilde{\mathbf{X}}, k)$.
5. Fix the rule $\mathcal{K}_{FUS}(X) : z = \arg\min_{i=1..k} \mathit{dist}(\tilde{X}, \tilde{C}_i)$, where $\tilde{X} = (x_{i1}x_{i2} \dots x_{im})'$ and $\tilde{C}_1, \dots, \tilde{C}_k \in \tilde{\mathcal{X}}$.

Fig. 4. Techniques for finding hidden contexts.

5.1 Evaluation criteria

To measure the three desired characteristics (independence from the class labels, validity and stability) we adopt metrics commonly used in clustering evaluation.

For **measuring independence** between the labels and the identified contexts, we employ *Normalized Mutual Information* (NMI), which is widely used to assess clustering performance [18]. It evaluates the purity of clusters with respect to class labels. For the context identification task *low* NMI is desired. For two random variables $\mathbf{y}$ and $\mathbf{z}$ $\mathit{NMI}(\mathbf{y},\mathbf{z}) = I(\mathbf{y},\mathbf{z})/\sqrt{H(\mathbf{y})H(\mathbf{z})}$, where $I(\mathbf{y},\mathbf{z})$ is the mutual information, $H(\mathbf{x})$ and $H(\mathbf{z})$ are the respective entropies. Note, that $NMI \in [0,1]$ and $NMI(\mathbf{y},\mathbf{y}) = 1$. Given the context assignments $\mathbf{z}$ and the respective class labels $\mathbf{y}$, the NMI is estimated [18] as

$$\mathit{NMI}(\mathbf{y},\mathbf{z}) = \frac{\Sigma_{l=1}^{k}\Sigma_{h=1}^{c} N_{lh}\log\frac{nN_{lh}}{n_l\hat{n}_h}}{\sqrt{(\Sigma_{l=1}^{k} n_l \log\frac{n_l}{n})(\Sigma_{h=1}^{c}\hat{n}_h\log\frac{\hat{n}_h}{n})}}, \qquad (1)$$

where n_l is the number of data instances contained in the cluster C_l $(1 \leq l \leq k)$, $\hat{n}_h$ is the number of instances belonging to the class h $(1 \leq h \leq c)$, N_{lh} is the number of instances that are in the intersection between the cluster C_l and the class h, and n is the total number of instances.

Measuring validity. If we optimized only NMI, assigning instances to contexts at random would be the optimal solution. To control that the identified contexts are not random, we require the identified context labels to be learnable from the data. We use the Naive Bayes (NB) classifier. $\mathit{VAL}(\mathbf{z}|\mathbf{X})$ is the error rate of NB using 10 fold cross validation, *the smaller* the better. We normalize it w.r.t. random assignment of contexts $\mathit{NVAL}(\mathbf{z}|\mathbf{X}) = \mathit{VAL}(\mathbf{z}|\mathbf{X})/\mathit{VAL}(\phi(k)|\mathbf{X})$, where $\phi(k)$ is a set of contexts (k) assigned at random, thus $\mathit{NVAL} \in [0,1]$.

Measuring stability. In addition to independence and validity we want to minimize the chance of overfitting the training data, which we measure using the stability index for clustering proposed in [13]. The dataset $\mathbf{X}$ is at random split into two sets of equal size $\{\mathbf{X_u} \cup \mathbf{X_v}\} = \mathbf{X}$. Each subset is clustered using the same clustering algorithm $\mathbf{u} = \mathit{clust}_u(\mathbf{X}_u)$, $\mathbf{v} = \mathit{clust}_v(\mathbf{X_v})$, $\mathit{clust}_u()$ and $\mathit{clust}_v()$ denotes fixed parameterizations resulting after clustering (e.g. cluster centers in k-means). Then the fixed $\mathit{clust}_v()$ is applied to the subset $\mathbf{X_u}$ to obtain alternative cluster assignment $\mathbf{\grave{u}} = \mathit{clust}_v(\mathbf{X_u})$. If clustering is stable, given a correct permutation $u^* = \mathit{map}(u)$ of cluster labels $\mathbf{\grave{u}}$ and $\mathbf{u}^*$ should be the same. The (in)stability index is the share of different cluster assignments $\mathit{STA}(\mathbf{u},\mathbf{\grave{u}}) = \frac{1}{n}\Sigma_{i=1}^{n}\mathbf{1}(u_i = \mathit{map}(\grave{u}_i))$, where $\mathbf{1}() = \mathbf{1}$ if true, otherwise 0. *The smaller* (in)stability (STA) the better. We normalize STA w.r.t. to random assignment to get $\mathit{NSTA} \in [0,1]$.

5.2 Datasets and experimental protocol

We test the techniques on thirty real classification datasets, which are diverse in size, dimensionality, number of classes and the underlying problems they represent. The characteristics are summarized in Table 1. We do not expect all of the

Table 1. Datasets: **N** - size, **d** - dimensionality, **y** - number of classes.

dataset	**N**	**d**	**y**	dataset	**N**	**d**	**y**	dataset	**N**	**d**	**y**
balance[1]	625	4	3	shuttle2[1]	14500	9	7	wis. cancer[1]	569	30	2
blood[1]	748	4	2	shuttle1[1]	43500	9	7	luxembourgh[4]	1901	31	2
mammographic[1]	961	5	2	vowels[1]	990	10	11	king-rock-pawn[1]	3196	36	2
car[1]	1728	6	4	page blocks[1]	5473	10	5	connect-4[1]	67557	42	3
elec2 [6]	44235	7	2	magic[1]	19020	10	2	marketing(c) [8]	8993	48	2
chess[4]	503	8	2	marketing(d) [8]	8993	13	2	brazil[3]	50000	49	2
pima[1]	768	8	2	adult[1]	32561	14	2	spam [8]	4601	57	2
nursery[1]	12960	8	5	australian[1]	690	15	2	ozone8[1]	2534	72	2
tic-tac-toe[1]	958	9	2	vehicles[1]	846	18	4	ozone1[1]	2536	72	2
contraceptive[1]	1473	9	3	german[1]	1000	24	2	user1[2]	1500	100	2

datasets to have distinct underlying contexts and we do not know the true number of contexts. Thus, we use the *stability* measure in the evaluation to indicate whether the found contexts are persistent in the data.

In the experiments we fix the number of contexts to $k = 3$ for all the datasets (no specific reason). Feature underselection technique requires to specify the number of features, we choose $m = 5$ for all the datasets. We normalize the feature values of the input features to fall in the interval [0,1], add 1% random noise and transform the data according to its principal components. Noise does not distort class discriminatory information, neither it influences the allocation of contexts. Noise and principal component rotation are needed to prevent ill posed covariance matrixes of some high dimensional datasets.

We empirically explore and compare five techniques: OVE, PRO, FUS, CLU and RAN. CLU is an ordinary clustering, which we use as the baseline method. Overlay (OVE), Projection (PRO) and Feature underselection (FUS) are the three context identification techniques introduced in this paper. RAN is a benchmark partitioning technique (sanity check), which assigns contexts uniformly at random. In this study we use k-means as the base clustering technique.

5.3 Results

Table 2 presents the results aggregated into three groups based on the dimensionality of the datasets: small (up to 10 features), medium (10-19 features) and large (more than 20 features) and all together. The results are plotted in Figure 5, where each dot represent one dataset. The figure shows that in terms of not capturing class labels (NMI) Overlay and Projection are doing well, while Feature underselection and the baseline Clustering are doing not that well. High NMI is consistent with higher validity, where Clustering outperforming the others, as presented in Table 2.

[1] UCI Irvine Machine Learning Repository `http://archive.ics.uci.edu/ml/`

[2] Katakis `http://mlkd.csd.auth.gr/concept_drift.html`

[3] PAKDD 2009 competition `http://sede.neurotech.com.br:443/PAKDD2009/`

[4] Žliobaitė collection `http://sites.google.com/site/zliobaite/resources-1`

Table 2. Summary of context identification results (the best in **bold**).

		CLU	OVE	PRO	FUS	RAN			CLU	OVE	PRO	FUS	RAN
all	NMI	0.12	**0.02**	0.05	0.03	0	small	NMI	0.13	**0.02**	0.06	0.05	0
	val.	0.12	0.18	0.18	**0.16**	1		val.	0.05	0.11	0.15	**0.08**	1
	stab.	0.36	**0.38**	0.59	0.53	1		stab.	0.43	**0.45**	0.63	0.49	1
large	NMI	0.09	0.01	**0.01**	0.01	0	med	NMI	0.13	0.05	0.11	**0.03**	0
	val.	0.19	0.26	**0.25**	0.27	1		val.	0.11	0.17	**0.11**	0.14	1
	sta.	0.32	**0.37**	0.50	0.67	1		sta.	0.29	**0.28**	0.68	0.38	1

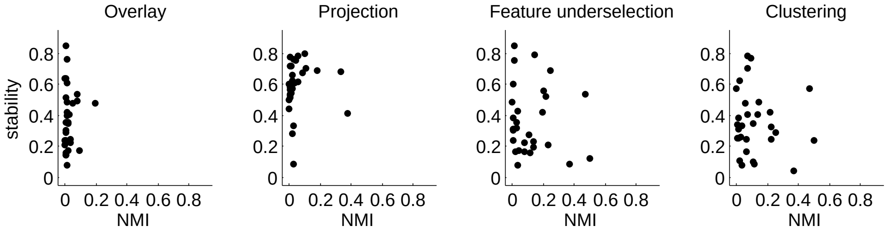

Fig. 5. Context identification results.

For all the techniques the validity deteriorates with increase in dimensionality. It can be expected, challenges of measuring distance in high dimensional space has been widely acknowledged [1]. FUS technique demonstrates the worst validity in high dimensional space. It can be explained by relatively low number of the selected features (we fixed $m = 5$).

Clustering has the best validity and stability, but captures a part of class discriminatory information, as expected, especially in low dimensional tasks. Projection has fine independence, good validity but it is rather unstable. This is mostly due to slicing of the resulting 1D projection. Likely, the resulting cut points might be not optimal and induce instability.

Feature underselection has surprisingly low stability as well as mediocre validity in high dimensional tasks. This is explainable by a fixed number of features m in our experiments (for comparability across datasets). In high dimensional spaces the selected features make rather small share of all features and are thus more likely to represent noise rather than context or predictive information. Good news is that for designing individual context handling strategies that can be resolved by manipulating m value.

Overlay has good independence and stability, while not so good but acceptable validity. This is due to supervised learning procedure to assign context to unseen instances. It introduces extra uncertainty, while the other techniques can identify context for an unseen instance directly.

To sum, all three techniques avoid capturing class label information well and show similar validity; in terms of stability, Overlay technique is preferable.

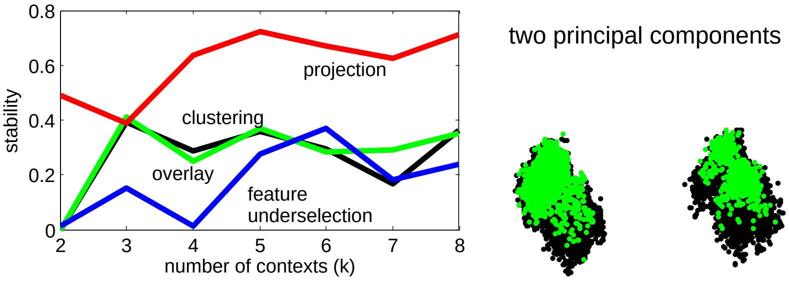

Fig. 6. Determining the number of contexts for *adult* data.

6 Case study

The following case study illustrates how context information can be used to benefit the final classification. We present it as a proof of concept rather than an attempt to select the most accurate context handling strategy. The case study focuses on determining the number of contexts (k) from the data and building one local classifier for each context.

We use *adult* dataset (alphabetically). We consider it suitable because of two reasons: the task (predicting income of a person) intuitively is context dependent and the dataset is relatively large ($>$ 30 th. instances).

We evaluate the accuracies using six base classifiers: decision tree (CART), logistic regression, Naive Bayes, linear discriminant (LDA), neural network (with 4 hidden layers), 1-nearest neighbor (1NN), and a collection. Collection means that the most accurate base classifier is selected from a pool of all but neural network (as it performs well on its own). We run 10-fold cross validation.

We run four context identification techniques (CLU, OVE, PRO, FUS) using different number of contexts $k = 2 \ldots 8$. The resulting stabilities are presented in Figure 6. Several strategies show the best stability at two contexts, then at four and seven. This tendency is also visible from the two principal components in the same figure. We choose to analyze $k = 4$ for this case study, since it is more interesting because of a larger distinction from single context. For a full picture, we also report the ranking of the techniques at $k = 2$ and $k = 7$.

How do we know that there are variable contexts at all? It can be concluded from the stability test and the plot of principal components. If there were no distinct contexts, the stability would be bad and the data in the principal component plot would be mixed.

Set up. The simplest context handling strategy is to build one local classifier for each context. We test how it works using the context labels identified by our techniques. For comparison we include a random split into contexts (RAN) and no split into contexts (ALL), which we use as baselines. We also add to the tests an ensemble (ENS) of CLU, OVE, PRO, FUS and RAN, which makes classification decision using simple majority voting.

Table 3. Errors of local classifiers.

	CLU	OVE	PRO	FUS	RAN	ALL	ENS
cart tree	19.62−	19.66−	19.60−	20.03−	20.44○	19.70	16.73●
log. reg.	16.51●	16.82●	17.43−	17.23●	17.45−	17.46	16.72●
n. bayes	19.28−	18.32●	19.02●	18.70●	19.31−	19.31	18.76●
LDA	21.62●	22.57●	22.30●	22.56●	23.28−	23.35	21.64●
neural n.	15.18●	15.79−	15.29●	15.83−	15.58●	15.91	15.06●
kNN	20.51−	20.62○	20.51−	20.54−	21.61○	20.50	20.47−
collection	19.86○	17.10●	18.71○	17.46−	17.52−	17.43	16.62●
mean	18.94	18.70	18.98	18.91	19.32	19.10	**18.00**

The testing errors are provided in Table 3. For the final evaluation, we average over the errors of different classifiers. Statistical significance is tested using a paired t-test. Symbol '●' means the technique is significantly better than the baseline (ALL). symbol '○' means the technique is significantly worse than the baseline. Symbol '−' means no statistical difference.

In terms of accuracy CLU performs not bad, NMI score shows that it captures not so much class label information on this data. Interestingly, RAN sometimes outperforms ALL. It can be seen as a variant of boosting, though suffering from small training sample. OVE and FUS performs on average better than CLU, it is mainly due to bad performance of CLU on the last test (collection).

We find that an ensemble (ENS) is the best in terms of accuracy. It is supported by experiments with different number of contexts. The rankings are:
$k = 2$ ENS$\prec$PRO$\prec$FUS$\prec$OVE$\prec$ALL$\prec$CLU$\prec$RAN;
$k = 4$ ENS$\prec$OVE$\prec$FUS$\prec$CLU$\prec$PRO$\prec$ALL$\prec$RAN;
$k = 7$ ENS$\prec$PRO$\prec$ALL$\prec$OVE$\prec$FUS$\prec$RAN$\prec$CLU.

The scope of the study is to analyze context identification rather than explore context handling strategies. Thus, we explore in depth only selection strategy and do not claim that it is the best. We report it as an illustration, complementary to the proposed identification techniques. It demonstrates, how the accuracy can be improved having no domain knowledge about underlying contexts, starting from identification of the number of contexts to training the actual classifiers.

7 Conclusion

Context identification techniques can be considered as a preprocessing step in classification, aimed to improve the accuracy, as well as contribute to understanding of the data. We require the contexts to be independent from the class labels, valid (non random) and stable.

We proposed three techniques for identifying hidden contexts from the data, directed not to capture class discriminatory information. The experiments on thirty datasets indicate that all the three techniques avoid capturing class label information pretty well and show similar validity; in terms of stability Overlay technique is preferable. The case study illustrates the benefits of context identification when used with classifier selection strategy.

Our study opens a range of follow up research opportunities for context handling strategies in static and dynamic (concept drift, discrimination aware learning) settings.

References

1. K. S. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is "nearest neighbor" meaningful? In *Proc. of the 7th Int. Conf. on Database Theory (ICDT'99)*, pages 217–235, 1999.
2. P. Brézillon. Context in problem solving: a survey. *Knowledge Engineering Review*, 14(1):47–80, 1999.
3. R. A. Dara, M. Makrehchi, and M. S. Kamel. Filter-based data partitioning for training multiple classifier systems. *IEEE Trans. on Knowledge and Data Engineering*, 22(4):508–522, 2010.
4. R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* Wiley-Interscience Publication, 2000.
5. D. Frosyniotis, A. Stafylopatis, and A. Likas. A divide-and-conquer method for multi-net classifiers. *Pattern Analysis and Applications*, 6(1):32–40, 2003.
6. M. Harries. Splice-2 comparative evaluation: Electricity pricing. Technical report, U. New South Wales, 1999.
7. M. Harries, C. Sammut, and K. Horn. Extracting hidden context. *Machine Learning*, 32(2):101–126, 1998.
8. T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: data mining, inference and prediction.* Springer, 2005.
9. I. Katakis, G. Tsoumakas, and I. Vlahavas. Tracking recurring contexts using ensemble classifiers: an application to email filtering. *Knowledge Information Systems*, 22(3):371–391, 2010.
10. M. Lim and S. Sohn. Cluster-based dynamic scoring model. *Expert Systems with Appl.*, 32(2):427–431, 2007.
11. R. Liu and B. Yuan. Multiple classifiers combination by clustering and selection. *Information Fusion*, 2(3):163–168, 2001.
12. J. Ren, X. Shi, W. Fan, and Ph. S. Yu. Type-independent correction of sample selection bias via structural discovery and re-balancing. In *Proc. of the SIAM Int. Conf. on Data Mining (SDM '08)*, pages 565–576, 2008.
13. V. Roth, T. Lange, M. Braun, and J. Buhmann. A resampling approach to cluster validation. In *Proc. of Int. Conf. on Computational Statistics*, pages 123–128, 2002.
14. T. Strang and C. Linnhoff-Popien. A context modeling survey. In *Workshop on Advanced Context Modelling, Reasoning and Management at the 6th Int. Conf. on Ubiquitous Computing (UbiComp 2004)*, 2004.
15. P. Turney. The identification of context-sensitive features: A formal definition of context for concept learning. In *Proc. of the ICML'96 Workshop on Learning in Context-Sensitive Domains*, pages 53–59, 1996.
16. P. Turney. The management of context-sensitive features: A review of strategies. In *Proc. of the ICML'96 Workshop on Learning in Context-Sensitive Domains*, pages 60–65, 1996.
17. G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.
18. M. Wu and B. Scholkopf. A local learning approach for clustering. In *Avances Neural Information Processing Systems (NIPS'06)*, 2006.