Under consideration for publication in Knowledge and Information Systems

Exploiting patterns to explain individual predictions

Yunzhe Jia[1], James Bailey[1], Kotagiri Ramamohanarao[1],
Christopher Leckie[1], Xingjun Ma[1]

{yunzhej@student. ,baileyj@, kotagiri@, caleckie@,
xingjun.ma@}unimelb.edu.au

[1]School of Computing and Information Systems,
University of Melbourne, Parkville, Victoria, Australia

Abstract. Users need to understand the predictions of a classifier, especially when decisions based on the predictions can have severe consequences. The explanation of a prediction reveals the reason why a classifier makes a certain prediction and it helps users to accept or reject the prediction with greater confidence. This paper proposes an explanation method called Pattern Aided Local Explanation (PALEX) to provide instance-level explanations for any classifier. PALEX takes a classifier, a test instance and a frequent pattern set summarizing the training data of the classifier as inputs, then outputs the supporting evidence that the classifier considers important for the prediction of the instance. To study the local behavior of a classifier in the vicinity of the test instance, PALEX uses the frequent pattern set from the training data as an extra input to guide generation of new synthetic samples in the vicinity of the test instance. Contrast patterns are also used in PALEX to identify locally discriminative features in the vicinity of a test instance. PALEX is particularly effective for scenarios where there exist multiple explanations. In our experiments, we compare PALEX to several state of the art explanation methods over a range of benchmark datasets and find that it can identify explanations with both high precision and high recall.

1. Introduction

Interpretability has been recognized as an essential requirement for the successful deployment of data mining techniques. One important aspect of interpretability is being able to provide explanations for the predictions of classifiers. In some domains such as medical diagnosis, marketing analysis and criminal analysis, accuracy is not the only concern when evaluating a model - the logical reasoning behind its predictions are also important and a classifier may make a correct prediction based on incorrect evidence. Knowing that a prediction is made based on reasonable evidence by the model, helps

users to understand the prediction better. Thus they can accept or reject the prediction with greater confidence. Moreover, explanations for predictions are mandatory in some cases, i.e., the European Union's new General Data Protection Regulation, which took effect in 2018, grants users the right to ask for an explanation of any algorithmic decision made about them, such as the refusal of a loan application (Goodman and Flaxman, 2016).

It is desirable that classifiers are both interpretable and accurate. However, in most cases users have to balance accuracy and interpretability when they select a model. The challenge comes from the fact that complex models (e.g., random forests and neural networks) tend to be more accurate but less interpretable, while simple models (e.g., decision trees and logistic regression) tend to be less accurate but more interpretable. This raises the problem: can we increase the interpretability of complex models without sacrificing their accuracy? In situations where interpretability is about the ability to provide explanations, we can use model-agnostic explanation methods.

Early works (Craven and Shavlik, 1996) (Kurfess, 2000) (Barakat and Diederich, 2005) (Fung, Sandilya and Rao, 2005) (Martens, Baesens, Van Gestel and Vanthienen, 2007) on providing explanations recognize the interpretability of patterns that are conjunctions of feature-value conditions, and apply pattern based methods to explain complex models, such as neural networks and SVMs, at the global model level. This category of explanations is called **global explanations**, as it involves descriptions of the whole model. However, global fidelity does not necessarily imply local fidelity (Ribeiro, Singh and Guestrin, 2016*c*), and global explanations may not be very useful for understanding an individual prediction.

Patterns have been widely adopted in the interpretability research community as they are intrinsically easy to interpret. Recent works (Robnik-Šikonja and Kononenko, 2008) (Adler, Falk, Friedler, Rybeck, Scheidegger, Smith and Venkatasubramanian, 2016) (Wang, Schaul, Hessel, Van Hasselt, Lanctot and De Freitas, 2016) (Ribeiro et al., 2016*c*) on instance-level **local explanations** for individual predictions investigate the feature contributions and generate a vector of weights as the form of an explanation. However, there are drawbacks to this type of method: (1) A representation using a vector of weights for an explanations is less precise, as the meaning of the weights can be ambiguous depending on the feature scales, which means that the same weight value could have different interpretations for different features that use different value ranges (i.e., when a feature is in the range of [0,100] and another feature in the range of [0, 50000]). (2) These methods only produce a single explanation for a prediction, and are not suitable for situations where there exist more than one reasonable explanation for a single prediction (e.g., in a disjunctive normal form where the result is true, there might be more than one true clause).

When extracting local explanations, existing methods typically generate a set of neighbour instances of a test instance, and then find local explanations from the generated neighbours given a classifier. The quality of the explanations depends heavily on the quality of the generated neighbourhood. In order to precisely find the local explanation for a test instance, the generated neighbour instances should be representative and contain sufficient information about the local behavior of the classifier (i.e., the local classification boundary). Unlike most techniques that use standard Euclidean distance to assess the proximity of a test instance to its neighbours, we explore the use of patterns to assess the proximity. Another key aspect is to extract explanations that reveal why the classifier makes a given prediction about the test instance. For this task, we employ contrast patterns to identify discriminative features for the given test instance.

This paper proposes a pattern aided method (PALEX) to find **supporting evidence** for the predictions of any classifier. The main contributions include:

- PALEX exploits frequent patterns for discovering local explanations for a test prediction. In particular, it uses frequent patterns to better guide the generation of a local neighbourhood around a test instance, rather than randomly generating it. This neighbourhood is then used as the basis for formulating explanations.
- PALEX extracts contrast patterns as explanations, finding the discriminative features for the class of an individual instance based on the local neighbourhood. It does not make any assumption about local linearity.
- PALEX enriches explanation power by providing multiple explanations for an individual prediction.

2. Preliminaries

Instance and dataset

An $instance$ is a set of attribute-value pairs. The domain of an attribute can be discrete (where the attribute is called nominal/categorical) or continuous (where the attribute is called numeric/continuous). One nominal attribute is selected as the $class/label$ of an instance. A $dataset$ consists of a set of instances, where all instances have the same attributes with possibly different values.

Classifier

A $classifier$ is a mapping relation $f(x)$ that maps an instance x (without class information) to a class label. Some classifiers are able to provide a confidence (or probability score) for the predictions.

Pattern

A $pattern$ is a conjunction of conditions. A condition may have one of three forms: i) $A = a$ or ii) $B \leq b$, or iii) $C > c$, where A is a nominal attribute, and B and C are numeric attributes.

Matching dataset

An instance x *matches* a pattern p if all conditions in p are true for x. The matching dataset of a pattern p in a dataset D is the subset of D where all instances in the subset match p, and is formally defined as $mds(p, D) = \{x \in D | x \text{ matches } p\}$.

Support

The support of a pattern p in a dataset D is the ratio of the size of the matching dataset to the size of D, and is defined as $supp(p, D) = \frac{|mds(p,D)|}{|D|}$.

Growth ratio

The growth ratio of a pattern p in dataset D_1 to dataset D_2 is the ratio of the corresponding supports, and is defined as $GrRatio(p, D_1, D_2) = \frac{supp(p,D_1)}{supp(p,D_2)}$. In particular,

$GrRatio(p, D_1, D_2) = \infty$ if $supp(p, D_2) = 0$ and such patterns are known as jumping emerging patterns (Fan and Ramamohanarao, 2006). D_1 and D_2 can be two datasets, or two subsets of a dataset (i.e., divided by class labels).

Frequent pattern

A pattern p is called a frequent pattern in dataset D if its support is greater than a user-specified threshold $minSupp$.

Contrast pattern

Given two datasets D_1 and D_2, a frequent pattern p is called a $contrast\ pattern$ when the growth ratio $GrRatio(p, D_1, D_2)$ is greater than a user-specified threshold $minRatio$, which means p occurs more frequently in D_1. Note that other metrics (e.g., support difference) can be used to define the contrast pattern. Interested readers can refer to (Dong and Bailey, 2012) for a discussion and comparison of the possible metrics.

Example. Taking the dataset D in Table 1 as an example, consider two patterns $p = \{F1 = true, F3 \leq 3\}, q = \{F2 = false\}$. The matching datasets of p, q are $mds(p, D) = \{x_1, x_2\}, mds(q, D) = \{x_1, x_3\}$, and the supports of both are $supp(p, D) = supp(q, D) = \frac{2}{4} = 0.5$. Both are frequent patterns if $minSupp < 0.5$. Assume the dataset is split into $D_1 = \{x_1, x_2, x_4\}$ and $D_2 = \{x_3\}$ by the class label, and the task is to find contrast patterns for D_1. Then the growth ratios of p, q are $GrRatio(p, D1, D2) = \frac{2/3}{0} = +\infty$, and $GrRatio(q, D1, D2) = \frac{1/3}{1/2} = \frac{2}{3}$. p is called a contrast pattern for D_1 and q is not, if $minRatio$ is set to be greater than $\frac{2}{3}$.

	F_1	F_2	F_3	class
x_1	$true$	$false$	1	1
x_2	$true$	$true$	2	1
x_3	$false$	$false$	5	0
x_4	$false$	$true$	3	1

Table 1. Dataset for demonstration of patterns.

3. Related Work

Pattern based interpretable models

One category of classifiers, called interpretable models, are inherently able to provide explanations for their predictions. Since patterns represented as sets of attribute-value pairs can be easily understood by users with or without a data mining background, pattern based classifiers (Caruana, Lou, Gehrke, Koch, Sturm and Elhadad, 2015) (Kohavi, 1995) (Letham, Rudin, McCormick, Madigan et al., 2015) (Wang and Rudin, 2015) are a promising class of data mining techniques that are easy to interpret. For example, the path from the root to the leaf of a decision tree for a given instance can be used to explain the prediction, or a matching rule in the rule list can also be used as the explanation. The goal of this research field is to develop models that are interpretable. For such models, there is always a trade-off between interpretability and accuracy (Freitas, 2014).

Global explanations

A common model-agnostic approach is to extract post-hoc explanations at the model-level (Ribeiro, Singh and Guestrin, 2016*a*). Early works trained pattern based models or extracted patterns to mimic black-box classifiers. Craven et al. (Craven and Shavlik, 1996) build decision trees and Kurfess (Kurfess, 2000) extract patterns to approximate neural networks. They re-label the training data using a trained network and then learn a decision tree or extract patterns on new training data. Similarly, Barakat et al. (Barakat and Diederich, 2005), Fung et al. (Fung et al., 2005) and Martens et al. (Martens et al., 2007) extract patterns for SVMs from the re-labeled training data. Instead of interpreting the behaviour of the whole classifier, SCaPE (Duivesteijn and Thaele, 2014) uses patterns to summarize the characteristics of the subgroup of data on which the classifier is likely to make incorrect predictions. GoldenEye++ (Henelius, Puolamäki, Karlsson, Zhao, Asker, Boström and Papapetrou, 2015) explains a classifier from the view of interacting attributes by grouping the attributes exploited by the classifier.

Explanations provided by these methods are model-level global explanations. As pointed out in (Ribeiro et al., 2016*c*), global fidelity does not imply local fidelity, as features that are important at the model level are not necessarily important at an instance level. Our method proposed in this paper, utilizes this interpretability of patterns which are easily understandable by humans, and identifies local explanations represented using contrast patterns at the instance level.

Local explanations

Recent works have focused on instance-level local explanations, which are explanations for individual predictions. Robnik et al. (Robnik-Šikonja and Kononenko, 2008), Henelius et al. (Henelius, Puolamäki, Boström, Asker and Papapetrou, 2014), Adler et al. (Adler et al., 2016), and (Koh and Liang, 2017) compute a vector of weights representing the predictive powers of attributes. Parzen (Baehrens, Schroeter, Harmeling, Kawanabe, Hansen and Müller, 2010) calculates the gradient of the predictor with respect to a given instance to explain its prediction, and Selvaraju et al. (Selvaraju, Cogswell, Das, Vedantam, Parikh and Batra, 2017), and Wang et al. (Wang et al., 2016) similarly compute the gradient as an image mask.

LIME (Ribeiro et al., 2016*c*) trains a local linear interpretable model in the vicinity of an instance and uses this new model to explain the instance. aLIME (Ribeiro, Singh and Guestrin, 2016*b*) (Ribeiro, Singh and Guestrin, 2018) is a variation of LIME, and is the first method to use patterns (which are referred to as "anchor" rules in aLIME) as local explanations. Both methods use extra data preprocessing steps (i.e., normalization) to calculate the distance function when assessing the vicinity of an instance. Our proposed method generates the neighbours using a kernel-like distance function such that no data preprocessing is needed. Moreover, unlike aLIME that uses a bottom-up construction procedure that favors high precision to discover pattern explanations, our method PALEX adopts contrast pattern mining techniques to generate contrast patterns as explanation candidates.

Importantly, most existing explanation systems only provide a single explanation for an individual prediction. In contrast, PALEX enriches explanations by providing multiple explanation candidates, which is useful in scenarios where multiple plausible explanations are reasonable. A comparison between PALEX and existing techniques is provided in Table 2.

Method	Multiple explanation support	Assumption that data is locally linearly separable	Representation of explanation	Categorical data support	Extra information requirement
Feature selection based (Robnik-Šikonja and Kononenko, 2008) (Henelius et al., 2014) (Adler et al., 2016) (Koh and Liang, 2017)	No	No	Vector of weights	Yes	No
Gradient-based (Baehrens et al., 2010) (Wang et al., 2016) (Selvaraju et al., 2017)	No	No	Vector of weights	No	No
LIME (Ribeiro et al., 2016*c*)	No	Yes	Vector of weights	Yes	No
aLIME (Ribeiro et al., 2018)	No	No	Pattern	Yes	No
PALEX (proposed)	Yes	No	Pattern	Yes	Frequent patterns from experts/training data

Table 2. Comparison of local explanation methods.

4. Patterns as Explanations

There are two common representations of explanations: *vectors of weights*, and *patterns*. When using vectors of weights as explanations, each weight corresponds to the predictive power of an attribute. A vector of weights is arguably more complex for a user to comprehend than a pattern, given the simpler format of patterns. Moreover, if there exist multiple explanations, a single vector of weights is unable to capture all the required information. Patterns also have the potential to provide meaningful value ranges for numeric attributes. Exploring meaningful value ranges is not considered in this paper, but this direction is an interesting area for future work. Next, we give a formal definition of patterns as explanations (similar to the definition in (Martens and Provost, 2011)).

First, we introduce the notation $\mathcal{G}(f, x, p)$ that takes a classifier f, an instance x and a pattern p as inputs (assume $f(x)$ gives the class label c), and outputs the probability that the prediction is still $c = f(x)$ if p is violated for x, and is defined as:

$$\mathcal{G}(f, x, p) = \mathbb{E}_{x' \in x \backslash p}[\mathbb{P}_f(c|x')] \tag{1}$$

where $\mathbb{E}[\cdot]$ calculates the expectation, $\mathbb{P}_f(c|x)$ is the probability that x is classified as c by f and $x \backslash p$ denotes all possible instances where all conditions in p are violated for x.

Assume a dataset consists of two features $F_1, F_2 \in \{v_1, v_2, v_3\}$ and class label $c \in \{0, 1\}$. Given an instance $x : F_1 = v_1, F_2 = v_1$, a pattern $p : (F_1 = v_1)$, and a classifier f that give prediction $f(x) = 1$ with probability $\mathbb{P}_f(c = 1|x) = 0.8$, there are two possible perturbations of x to violate p such that $x \backslash p = \{x_1, x_2\}$ where $x_1 : F_1 = v_2, F_2 = v_1$ and $x_2 : F_1 = v_3, F_2 = v_1$. Suppose x_1, x_2 are equally likely distributed, and $\mathbb{P}_f(c = 1|x_1) = 0.4$, $\mathbb{P}_f(c = 1|x_2) = 0.3$, then $\mathcal{G}(f, x, p) = \mathbb{E}_{x' \in \{x_1, x_2\}}[\mathbb{P}_f(c = 1|x')] = \frac{1}{2}\mathbb{P}_f(c = 1|x_1) + \frac{1}{2}\mathbb{P}_f(c = 1|x_2)$ = 0.35, meaning that the probability of being classified as $c = f(x)$ is 0.35 if x is violated by p.

Exact calculation of $\mathbb{E}_{x' \in x \backslash p}[\mathbb{P}_f(c|x')]$ is usually computationally infeasible (espe-

cially for numeric features when there are an infinite number of instances in $x \backslash p$), thus the approximation is computed by $\mathcal{G}(f, x, p) = \mathbb{E}_{x' \in x \backslash p}[\mathbb{P}_f(c|x')] \approx \frac{1}{M} \sum_{i=1}^{M} \mathbb{P}_f(c|z_i)$. Here z_i is a generated sample where feature values are the same as x for features that do not appear in pattern p, and feature values are randomly chosen to violate p for features that appear in p.

Next, we can define the notation of explanations using patterns. Given an instance x and a classifier f, a pattern p is called an explanation for $f(x) = c$ if:

1. x matches p,
2. $\mathbb{P}_f(c|x) > \mathcal{G}(f, x, p)$.

The first property ensures that the explanation contains correct attribute values from x. The second property says that if a pattern is violated, the probability of being predicted as the same class label should drop. In other words, the explanation should be the supporting evidence for the prediction.

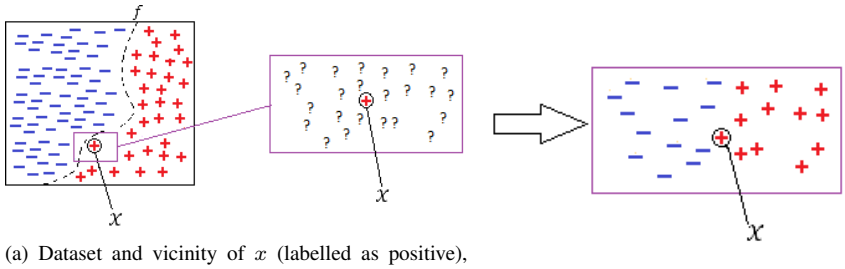
5. Proposed Algorithm: PALEX

We next describe our proposed method - Pattern Aided Local Explanations (PALEX). PALEX discovers a set of contrast patterns as the explanations for an individual prediction. In order to study the behavior of a given classifier in the vicinity of the test instance being predicted, PALEX initially generates new neighbouring samples for the instance. PALEX uses frequent patterns to guide the generation of these neighbours, according to the intuition that valid neighbours must contain matching frequent patterns in order to be close to the test instance. Closeness is measured using a distance function in pattern space. Then these generated instances are labelled by the given classifier: instances with the same label as the testing instance are treated as the positive set and the rest as the negative set. Contrast patterns for the positive set against the negative set are then mined as explanations, and these patterns capture the discriminative features for the testing instance against its neighbours with a different class label.

PALEX differs from LIME in two key aspects: (1) Unlike LIME which uses Euclidean distance, PALEX uses the input frequent patterns to generate a more precise neighbourhood for the given instance. (2) Instead of using an interpretable model to mimic the local behavior, PALEX applies a contrast pattern mining technique to capture locally important information. Comparing with aLIME (Ribeiro et al., 2018), a variant of LIME, PALEX does not require successful detection of the local boundary, whereas for aLIME, a poorly detected boundary may result in degraded performance. aLIME employs a bottom-up strategy to identify the explanation pattern and the construction process is targeted towards high precision scenarios. PALEX selects the explanation pattern(s) from a set of easily mined candidates (contrast patterns) and is targeted towards optimising both precision and recall.

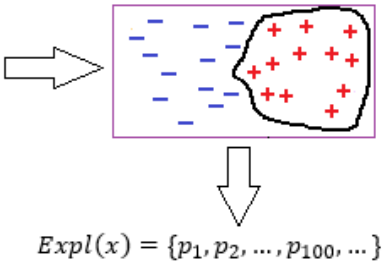
The number of explanations found by contrast pattern mining could be very large, and needs to be pruned. An additional step to extract the top K representative explanations is carried out in the final phase. We formulate this process as a linear optimization problem with multiple constraints.

The steps of PALEX are shown in Algorithm 1, and graphically illustrated in Figure 1.

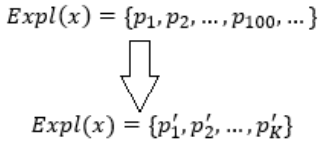


(a) Dataset and vicinity of x (labelled as positive), question marks (?) represent generated neighbours of x. (Step 1)

(b) Newly generated neighbours are then labeled by the given classifier. (Step 2)



(c) Contrast patterns $\{p_1, p_2, \ldots\}$ for the instances with the same label as x are mined such that these patterns occur more frequently in the instances with the same label as x (which is labelled as positive) than in the rest.(Step 3)



(d) K contrast patterns $\{p'_1, p'_2, \ldots, p'_K\}$ are chosen as final outputs with a pattern selection process. (Step 4)

Fig. 1. Overview of PALEX process

Algorithm 1: PALEX

Input: classifier f, instance x, frequent pattern set PS, number of samples N, minimal support $minSupp$, minimal growth ratio $minRatio$, number of explanations K

Output: K contrast patterns as explanations

$nei = generateNeighbours(x, PS, N)$;
for z *in* nei **do**
  $z.label = f(z)$
end
$cp = mineCPs(nei, minSupp, minRatio, x.label)$;
remove contrast patterns in cp that do not satisfy the explanation definition properties in Section 4;
$explanations = selectExplanations(cp, K, f)$ (Solve Equation 7);
return $explanations$;

5.1. Methodology

Input

The inputs for PALEX include:(1) **classifier** f, which can be any given classifier and it is treated as a black box. The explanation process requires it to provide probabilities (or similar scores) for the predictions; (2) **instance** x, which is the instance whose predic-

Algorithm 2: generateNeighbours

Input: instance x, frequent pattern set PS, number of samples N
Output: N instances

```
nei = {};
while nei.size < N do
    z = x.copy();
    uniformly select a subset of features z and uniformly assign a possible value
      to each chosen feature.
    z.weight = exp^{-(Dist(x,z|PS))^2} (where Dist(x, z|PS) is defined in
      Equations 2 and 3);
    nei = nei ∪ {z}
end
return nei;
```

tion by f needs to be explained; (3) **frequent pattern set** PS, which can be considered as a summary or compression of the data. It can be mined from the data used to train f, or provided by the users based on their domain expertise; (4) **number of samples** N, which is the number of neighbours generated in the vicinity of x. A large value of N can slow down the contrast pattern mining step. On the other hand, a small value of N is not sufficient to simulate the behavior of f in the vicinity of x; (5) **minimal support** $minSupp$ and (6) **minimal growth ratio** $minRatio$ are the thresholds used to mine contrast patterns. If these two values are set too large, the number of contrast patterns mined may be too small and thus useful explanations could be ignored. On the other hand, if they are set too small, there will be numerous contrast patterns, which makes it harder to choose representatives; (7) **number of explanations** K, which is the threshold for the number of explanations such that at most K explanations are returned.

Output

K contrast patterns satisfying the definition (Section 4) are returned.

Process

The process of PALEX is as follows.

First, N neighbours of x are generated (described in Algorithm 1). PALEX perturbs x to generate new instances by random perturbation such that at each generation for each feature it tosses a coin to decide whether it should be perturbed or not, and if yes, assigns a possible value (which can be any item for a categorical feature, or any value in the domain range for a numerical feature) to the feature. Each newly generated instance z is weighted by $\exp^{-(Dist(x,z|PS))^2}$, where $Dist(x, z|PS)$ (see Equations 2 and 3) is the distance (in frequent pattern space based on PS) of z to x.

Second, the class labels of the newly generated samples are predicted by f.

Third, contrast patterns that occur frequently in the samples with the same class label as x and infrequently in other classes are extracted. The pattern mining algorithm should be able to handle weighted instances, as the neighbours are weighted by their distances to the test instance. For example, a common strategy is to calculate the weighted support by $supp(p, D) = \frac{\sum_{x_j \in mds(p,D)} x_j.weight}{\sum_{x_i \in D} x_i.weight}$, where $mds(p, D)$ is the matching dataset of p in D. For example, given a dataset $D = \{x_1, x_2, x_3\}$ where the instances are weighted

by $x_1.weight = 0.1, x_2.weight = 0.2, x_3.weight = 0.3$, and a pattern p that matches x_1 and x_2, the weighted support of p is $supp(p, D) = \frac{x_1.weight+x_2.weight}{x_1.weight+x_2.weight+x_3.weight} = 0.5$. Most contrast pattern mining methods that are able to deal with instance weights can generate good explanation candidates. For efficiency purposes, we use a random forest based mining process that was proposed in (Shang, Tong, Peng and Han, 2016) (similar to (Kang and Ramamohanarao, 2014)), and one can also replace this with other mining methods that have similar performance with the same thresholds. For other contrast pattern mining algorithms, interested readers can refer to (Dong and Bailey, 2012).

Lastly, K patterns are selected from the set of contrast patterns. Patterns that do not meet the definition of an explanation are removed, then at most K contrast patterns are selected with an optimization step (outlined in Section 5.3) from the remaining set.

5.2. Sampling in the frequent pattern space

We assign weights to the newly generated instances using a distance function that measures the distances of these instances to the given test instance, such that a closer instance is assigned a higher weight, making it more influential in the explanation process that follows.

Given a frequent pattern set $PS = \{p_1, p_2, \ldots, p_M\}$, an instance x is mapped to an M-dimensional numeric vector x', where the i-th feature is non-zero if x matches p_i, representing the strength of p_i (support is used in this paper). The distance between two instances x, z is measured by $D(x', z')$, where x', z' are their projections in the frequent pattern space. $D(x', z')$ can be any traditional distance measure and l_1 distance is used in this paper. Formally,

$$Dist(x, z|PS) = D(x', z') = \sum_i |x'_i - z'_i| \tag{2}$$

where $x' = (x'_1, \ldots, x'_M)$ such that

$$x'_i = \phi(x_i) = \begin{cases} supp(p_i), & x \text{ matches } p_i \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

and z' is defined similarly.

This technique with a feature map $\Phi(x) = (\phi(x_1), \ldots, \phi(x_M))$ is similar to a kernel method that maps the original features into a certain feature space. Frequent patterns mined from the training data (or obtained from experts with background knowledge) contain information about important correlations. When generating a neighbourhood (vicinity) around the instance, it is important to preserve correlations (frequent patterns) possessed by the test instance. Intuitively speaking, a frequent pattern that is shared between the test instance and its neighbours acts as a connection between them. The more connections two instances have, conceptually the closer they are (like in a social network the more mutual friends two people have, the higher the probability that they are friends). Measuring distance in the frequent pattern space calculates the connections (shared correlations) between instances. More discussions about pattern based feature transformation can be found at (Jia, Bailey, Kotagiri and Leckie, 2018).

5.3. Explanation selection

This process requires the classifier f to be able to provide probabilities or similar confidence scores for its predictions in order to evaluate the power of a contrast pattern as an explanation. The scoring function of contrast pattern p is defined as

$$score(p|x, f) = \mathbb{P}_f(c|x) - \mathcal{G}(f, x, p) = \mathbb{P}_f(c|x) - \mathbb{E}_{x' \in x \setminus p}[\mathbb{P}_f(c|x')] \tag{4}$$

where $c = f(x)$ is the prediction of x by f, and $\mathbb{P}_f(c|x)$ is the corresponding probability. $\mathcal{G}(f, x, p)$ is the probability that the prediction is still c if p is violated (see Equation 1). Note that the scoring function is different from the support or growth ratio of a pattern, as the scoring function measures the prediction probability/confidence change if the pattern is violated, while the support or growth ratio is an intrinsic property of the pattern in the data.

Given contrast patterns CP mined from Step 3, the choice of K of them as explanations is formulated as the following optimization problem:

$$\begin{aligned} &expl(x) = \underset{P \subseteq CP}{\operatorname{argmax}}\{L(f, P) - \alpha\Omega(P)\} \\ &\text{subject to } |P| \leq K \end{aligned} \tag{5}$$

where $L(f, P) = \sum_{p \in P} supp(p) * score(p|x, f)$ which is a weighted summation of the pattern score, α is the regularization rate (we set it to 1 as the experiments suggest that it achieves best performance on average when multiple explanations are required) and $\Omega(P)$ computes the average overlap of pair-wise patterns as the penalty term. We use the Jaccard similarity, which is defined as

$$\Omega(P) = \sum_{p_i, p_j \in P, i \neq j} \frac{|mds(p_i) \cap mds(p_j)|}{|mds(p_i) \cup mds(p_j)|}. \tag{6}$$

Other overlap measuring methods can also be used. The overlap constraint forces the selected patterns to be as diverse as possible.

This optimization problem can be re-formulated as a MIP (Mixed Integer Programming) problem and solved by a stochastic MIP solver. It is re-formulated as:

$$\begin{aligned} \max_{a_i, b_{ij}} \quad & a_i * score_i - b_{ij} * penalty_{ij} \\ \text{subject to} \quad & \sum_i a_i \leq K \\ & a_i + a_j - 2b_{ij} \leq 1, \forall i, j \\ & a_i, b_{ij} \in \{0, 1\} \end{aligned} \tag{7}$$

where a_i, b_{ij} are the decision variables, $a_i = 1$ if pattern p_i is chosen and $b_{ij} = 1$ if patterns p_i, p_j are chosen, $score_i = score(p_i|x, f)$ and $penalty_{ij} = \frac{|mds(p_i) \cap mds(p_j)|}{|mds(p_i) \cup mds(p_j)|}$. The first constraint makes sure that at most K patterns are selected. The second constraint defines the relation of a_i and b_{ij}, and the last constraint defines the type of the decision variables. Users can also add other optional constraints of their own choice. We have tried constraints in terms of pattern length, support and ratio, but there is no significant improvement.

In particular, if $K = 1$ and only one explanation is required, the problem is transformed to find the explanation with the highest weighted score in Equation 4.

Method	Explanation
LIME	(0.29, 0.28, 0.51, -0.47)
aLIME	{size = SMALL, act = STRETCH}
PALEX	{color = YELLOW, size = SMALL}

(a) Single explanation case

Method	Explanation
LIME	(0.29, 0.26, 0.52, 0.49)
aLIME	{size = SMALL, act = STRETCH}
PALEX	{color = YELLOW, size = SMALL} OR {act = STRETCH, age = ADULT}

(b) Multiple explanations case

Fig. 2. Comparison of explanations generated by LIME, aLIME and PALEX.

5.4. Example

The example is adopted from the UCI balloon data with a custom classification rule. The dataset D consists of four features: color, size, act, age that are denoted as x_1, x_2, x_3, x_4 respectively, such that x_1 (color) $\in$ {YELLOW, PURPLE}, x_2 (size) $\in$ {SMALL, LARGE}, x_3(act) $\in$ {STRETCH, DIP}, x_4 (age) $\in$ {ADULT, CHILD}, and the class label c $\in \{T, F\}$. Assume the underlying classification model f is:

$$f(x) = \begin{cases} T, & (x_1\text{=YELLOW} \wedge x_2\text{=SMALL}) \vee (x_3\text{=STRETCH} \wedge x_4\text{=ADULT}) \\ F, & \text{otherwise} \end{cases}$$

Two test instances x_1 and x_2 are shown in Table 3. Three methods PALEX, LIME and aLIME are used to generate explanations for the predictions of x_1, x_2 by the underlying model f while treating f as a black box.

	color	size	act	age
x_1	YELLOW	SMALL	STRETCH	CHILD
x_2	YELLOW	SMALL	STRETCH	ADULT

Table 3. Test instances to illustrate explanations.

Single explanation case

Instance x_1 is used to test the single explanation scenario. The results are shown in Figure 2a. For instance x_1, it can be inferred from the model f that the true explanation is "because color is YELLOW and size is SMALL". LIME gives a vector of weights representing the contributions/importance of each feature, and it gives higher importance to features *act* and *age* which are not precise. aLIME gives an explanation that includes feature *act* but ignores feature *color*. PALEX provides an explanation that matches the true explanation.

Multiple explanations case

One advantage of PALEX is its ability to generate multiple explanations. Not only does it enable users to choose the best explanations from a small number of candidates, but it also naturally fits the scenario where there exist multiple true explanations. Instance

x_2 is used for testing the multiple explanations scenario, the results are shown in Figure 2b. Given the classification rule and the instance, it is clear that the two possible explanations are "because color is YELLOW and size is SMALL" or "act is STRETCH and age is ADULT". LIME gives a single explanation suggesting all features are important, but fails to capture the underlying truth, while aLIME gives a single explanation saying "because color is YELLOW and act is STRETCH". In contrast, PALEX generates both of the two explanations.

Multiple explanations can be useful in the scenario where the users want to generate exploratory insights from a model. Each explanation/insight can reveal some logical reasoning ideas from the model, and it is left to the users to verify whether it holds in reality or not.

5.5. Complexity analysis

PALEX consists of three major steps: synthetic neighbour generation, contrast pattern mining and pattern selection. Neighbour generation iteratively computes distances in the pattern space for N neighbours, and its complexity is $O(NL)$, where $L = |PS|$ is the size of the pattern set and will be influenced by data dimensionality if the pattern set is mined directly from the data. For contrast pattern mining, it is desirable to employ an efficient mining process. We use the random forest based mining method whose complexity is $O(TNd)$, where T is the number of trees and d is the maximum tree depth. For the pattern selection process, solvers normally take polynomial time in terms of the number of variables and constraints, thus the complexity is $O(S^{2c})$, where S is the number of contrast patterns generated by the previous step, and c is some positive number (e.g., 2.5 for Karmarkar's algorithm (Karmarkar, 1984)). Overall, the total complexity is $O(N(L + Td) + S^{2c})$. An important challenge in scaling for large datasets is that a frequent pattern set summarizing the training data is required as input, but since this mining process happens only once, existing efficient pattern mining methods for big data (see (Aggarwal and Han, 2014)) can be applied. Moreover, this problem can be avoided if the frequent patterns come from expert knowledge.

6. Experiments

In this section, we evaluate the quality of explanations in order to answer the question of whether the explanations are faithful to the model. The core idea of measuring the faithfulness of the explanations is to compare them with the true ones that are completely faithful to the model. In our experiments, the true explanations are either pre-defined using known generation of synthetic data, or extracted by looking inside an interpretable model (decision trees are used in this paper). The impact of the explanation selection process is also discussed.

6.1. Metrics

We adopt the evaluation metrics from LIME (Ribeiro et al., 2016*c*). Given a true explanation e_{true}, a D-dimensional instance x and a model f whose explanation for x is e, the faithfulness of the explanation is measured by the F_1 score:

$$F_1 = 2\frac{precision * recall}{precision + recall} \tag{8}$$

where $precision(e, e_{true}) = \frac{\sum_{d \in D} |e_d \times e_{true,d}|}{\sum_{d \in D} |e_d|}$ represents the fraction of features occurring in the explanation that are true, and $recall(e, e_{true}) = \frac{\sum_{d \in D} |e_d \times e_{true,d}|}{\sum_{d} |e_{true,d}|}$ represents the fraction of true features that the explanation covers.

6.2. Experimental setup

The experiments are conducted on four synthetic datasets and ten UCI datasets. For the synthetic data, the oracle classifiers and true explanations are predefined. For UCI data, decision trees are built on training data and the true explanations are extracted from the structures (prediction paths) of the trees. The averaged metrics over the testing data are calculated.

Benchmarks. Our method PALEX is compared with *LIME*, a local explanation system that computes the contributions of features for an individual prediction, *aLIME* (a variation of LIME), and a global explanation method (denoted as *Global*). For LIME, a set of features with values that are in favor of the prediction and with a contribution (weight) greater than 0.01 is extracted to form an explanation pattern (contrast pattern) that supports the prediction. For aLIME, the anchor rule generated is used as the explanation. The parameters used for LIME and aLIME are chosen via a validation process (similar to the process described later for PALEX) for optimal performance. For Global (similar to (Koh and Liang, 2017)), the training data is relabeled by the given classifier, and a set of classification rules are generated, then the matching rule is selected as the explanation.

Parameter settings In our experiments, for PALEX, the initial frequent pattern set is obtained from the training data via FP-growth (Han, Pei and Yin, 2000) with minimal support = 0.1. Parameter optimization via a nested validation process during the training process is a common way for classification tasks, however, it is difficult for explanation systems due to the fact that true model explanations are always unavailable when deploying such systems in real-life problems. A solution to decide the parameters is to generate a parameter setting using validation datasets whose true model explanations are available, and then apply the same setting for any new dataset. To simulate such a process, we apply PALEX with different parameter settings of three major parameters N, $minSupp$ and $minRatio$ on four separate UCI datasets (*adult*, *crx*, *hepatitis* and *ILPD*, which are different from those in Table 6 used for evaluation) as validation datasets, then the recommended parameter setting (that achieves the best F_1 score on average over the validation datasets) is used for all the datasets whose results are reported for comparison and analysis in this section. More specifically, each validation dataset is randomly split into two groups such that 80% is for training as T_1 and 20% is for test as T_2. Then grid search is used to find the optimal parameter setting. Every possible combination $(N, minSupp, minRatio)$ where $N \in \{50, 200, 500, 1000, 3000, 5000\}, minSupp \in \{0.1, 0.15, 0.2, 0.3, 0.4\}, minSupp \in \{0.1, 0.15, 0.2, 0.3, 0.4\}$ (the possible values are commonly used in pattern based methods) is tried on T_1 and evaluated on T_2 for all four validation datasets. There are 150 possible combinations, and the combination $(N^*, minSupp^*, minRatio^*)$ with best performance on average is chosen as the parameter setting for the rest of our experiments. The parameter setting that we find is $N^* = 500, minSupp^* = 0.1, minRatio^* = 5$. Similarly, the parameters for the benchmark methods are obtained via the same validation datasets, and a fixed setting is used for all other test data.

For the choice of the number of explanations K in PALEX, unless otherwise specified, we set it to 1 for evaluation purposes as all baselines generate only one explanation.

When deploying the PALEX system in real-life problems, users can employ their own validation dataset (i.e., one can use explanations that come from domain experts to approximate true model explanations) to repeat the process of generating parameter settings.

6.3. Faithfulness to oracle on synthetic data

We first conduct experiments on synthetic data. The advantage of using synthetic data is that we can test the explanation systems with an oracle classifier and easily obtain the true explanations. Four synthetic datasets are generated using the following DNFs:

1. $(f_1 \wedge f_2 \wedge f_3) \vee (\neg f_1 \wedge f_4 \wedge f_5)$, where $f_1, \ldots, f_5$ are binary attributes.
2. $((f_1 == 1) \wedge f_2 \wedge f_3) \vee ((f_1 == 2) \wedge f_4 \wedge f_5) \vee ((f_1 == 3) \wedge f_6 \wedge f_7) \vee ((f_1 == 4) \wedge f_8 \wedge f_9)$, where f_1 is numeric and $f_2, \ldots, f_8$ are binary attributes.
3. $(f_1 \wedge f_2) \vee (f_3 \wedge f_4) \vee (f_5 \wedge f_6) \vee (f_7 \wedge f_8)$, where $f_1, \ldots, f_8$ are binary attributes.
4. $(f_1 \wedge f_2 \wedge f_3) \vee (f_4 \wedge f_5 \wedge f_6) \vee (f_7 \wedge f_8 \wedge f_9)$, where $f_1, \ldots, f_9$ are binary attributes.

For each generation of the synthetic datasets, the feature values are randomly chosen and the class label is $T(true)$ if the corresponding DNF is satisfied. As the DNFs are used to construct the oracle classifier, no training data is required. Since the negation of a DNF rule is also a DNF, it is sufficient to examine the explanations for instances with label T.

Synthetic-1 and Synthetic-2 are used to test the *single-explanation* case and the other two datasets are used for the *multiple-explanation* case (where the number of generated explanations K is set to 2). The true explanation for an instance with label T is the set of features that occur in the clause satisfied by the instance. For example, assume the rule is $(f_1 \bigwedge f_2) \bigvee (f_3 \bigwedge f_4)$, for instance $(f_1 = true, f_2 = true, f_3 = false, f_4 = true)$, the satisfying clause is $(f_1 \bigwedge f_2)$, thus the true explanation is $\{f_1 = true, f_2 = true\}$. Similarly, for instance $(f_1 = true, f_2 = true, f_3 = true, f_4 = true)$, there are two possible explanations $\{f_1 = true, f_2 = true\}$ or $\{f_3 = true, f_4 = true\}$.

Results and discussion. For the multiple-explanation case, results are the average value of all the possible explanations. The results are reported in Table 4 and Table 5, and the results show that PALEX is more faithful to the oracle classifier.

	Precision				Recall			
Dataset	PALEX	LIME	aLIME	Global	PALEX	LIME	aLIME	Global
Syn-1	**1.00**	0.44	0.89	0.65	**1.00**	**1.00**	0.65	0.87
Syn-2	**0.99**	0.50	0.48	0.44	**0.82**	0.74	0.27	0.77
Syn-3	**0.87**	0.48	0.83	0.65	0.94	**1.00**	0.83	0.88
Syn-4	**1.00**	0.54	**1.00**	0.61	0.76	**0.80**	0.47	0.69

Table 4. Precision and recall results for synthetic data.

	F_1 (faithfulness)			
Dataset	PALEX	LIME	aLIME	Global
Syn-1	**1.00**	0.68	0.74	0.72
Syn-2	**0.86**	0.59	0.34	0.64
Syn-3	**0.90**	0.64	0.83	0.76
Syn-4	**0.83**	0.70	0.62	0.63

Table 5. F_1 (faithfulness) resutls for synthetic data.

6.4. Faithfulness to decision trees on UCI data

We conduct experiments on a variety of UCI datasets. The true explanations are extracted in the same way as LIME: decision trees are trained from the training data, and the true explanations for the test data are obtained from the trees, though the explanation systems treat the decision trees as black boxes.

Data. The datasets used are described in Table 6. Each dataset is randomly split into two groups such that 80% is used to train the classifier and the remaining 20% is used to evaluate the explanations.

Dataset	#inst	#nomAttr	#numAttr	#classes
Balloon	16	4	0	2
Blood	758	0	4	2
Breast-cancer	596	8	0	2
Diabetes	768	0	8	2
Ionosphere	351	0	34	2
Iris	150	0	4	3
Labor	57	8	8	2
Musk	6598	0	168	2
Titanic	3772	22	7	2
Vote	435	16	0	2

#inst - number of instances, #nomAttr - number of nominal attributes,
#numAttr - number of numeric attributes, #classes - number of classes

Table 6. UCI datasets description.

Classifiers and true explanations. Decision trees are trained for all datasets. The true explanation for the prediction of an instance is the set of features that occur in the path from the root to the leaf corresponding to the prediction.

Results and discussion. The metrics averaged over all test instances for each dataset are reported in Table 7 and Table 8. For precision, PALEX is able to achieve the best results in 7 out of 10 datasets. For recall scores, PALEX still wins in 6 out of 10 datasets and achieves comparable results with LIME and aLIME in the others. In terms of the F_1 score, PALEX wins in 8 out of 10 datasets.

	Precision				Recall			
Dataset	PALEX	aLIME	LIME	Global	PALEX	aLIME	LIME	Global
Balloon	**1.00(0.00)**	**1.00(0.00)**	0.5(0.00)	0.5(0.00)	0.88(0.22)	0.88(0.22)	**1.00(0.00)**	0.75(0.18)
Blood	**0.97(0.12)**	0.88(0.33)	0.78(0.25)	0.47(0.14)	0.72(0.28)	0.67(0.35)	**0.78(0.24)**	0.72(0.21)
Breast-cancer	0.96(0.18)	**1.00(0.00)**	0.61(0.26)	0.58(22)	0.92(0.23)	**0.92(0.18**	0.90(0.25)	0.88(0.14)
Diabetes	**0.98(0.08)**	0.97(0.13)	0.79(0.27)	0.67(0.20)	**0.82(0.19)**	0.70(0.22)	0.58(0.28)	0.78(0.16)
Iris	**1.00(0.00)**	**1.00(0.00)**	0.83(0.37)	0.64(0.28)	**1.00(0.00)**	**1.00(0.00)**	0.83(0.37)	0.89(0.14)
Ionosphere	**0.90(0.21)**	0.83(0.29)	0.21(0.16)	0.44(0.25)	0.57(0.24)	0.55(0.26)	0.33(0.34)	**0.79(0.32)**
Labor	0.96(0.14)	**1.00(0.00)**	0.85(0.23)	0.68(0.15)	**0.97(0.09)**	0.97(0.10)	0.73(0.09)	0.77(0.06)
Musk	**0.91(0.12)**	0.84(0.18)	0.79(0.14)	0.56(0.15)	**0.81(0.18)**	0.64(0.22)	0.75(0.11)	0.68(0.33)
Titanic	0.78(0.18)	**0.96(0.16)**	0.62(0.28)	0.58(0.15)	**0.82(0.22)**	0.64(0.26)	0.50(0.24)	0.69(0.21)
Vote	**0.99(0.08)**	0.74(0.32)	0.83(0.26)	0.55(0.24)	**0.92(0.17)**	0.69(0.22)	0.56(0.24)	0.62(0.17)

Table 7. Precision and recall results of PALEX, aLIME, LIME and Global on UCI datasets (the best results are highlighted in bold, and standard deviations are included in parentheses)

	F_1 (faithfulness)			
Dataset	PALEX	aLIME	LIME	Global
Balloon	**0.92(0.14)**	**0.92(0.14)**	0.67(0.00)	0.59(0.12)
Blood	**0.79(0.20)**	0.74(0.32)	0.72(0.14)	0.52(0.20)
Breast-cancer	**0.94(0.20)**	0.92(0.12)	0.70(0.22)	0.77(0.22)
Diabetes	**0.85(0.12)**	0.79(0.17)	0.63(0.23)	0.68(0.24)
Iris	**1.00(0.00)**	**1.00(0.00)**	0.83(0.37)	0.74(0.24)
Ionosphere	0.63(0.15)	**0.65(0.27)**	0.22(0.14)	0.54(0.28)
Labor	0.96(0.10)	**0.98(0.05)**	0.88(0.16)	0.72(0.09)
Musk	**0.86(0.12)**	0.72(0.21)	0.76(0.15)	0.64(0.31)
Titanic	**0.77(0.13)**	0.73(0.21)	0.52(0.23)	0.61(0.19)
Vote	**0.94(0.12)**	0.70(0.26)	0.84(0.23)	0.59(0.11)

Table 8. F_1 (faithfulness) results of PALEX, aLIME, LIME and Global on UCI datasets (the best results are highlighted in bold, and standard deviations are included in parentheses)

6.5. Impact of sampling method

Gradient based sampling techniques have been widely used in the field of adversarial classification, where nearby instances with the opposite class label to a given instance are generated. We compare a gradient based sampling method with the proposed pattern based one. In the gradient based sampling method, the objective function is defined as $g(x') = dist(x, x') + L(Pr(y|x), Pr(y|x'))$, where $dist(x, x')$ is the cost of modifying x to x' (l_2 distance is used) and $L(Pr(y|x), Pr(y|x')$ measures the prediction probability difference (l_1 distance is used). To find the instance of the opposite class with minimal cost, the given instance x is moved towards the direction $-\lambda\Delta g(x')$ where λ is the step size. Because the gradient based sampling method always generates samples along one direction towards a local optima, we run the methods several times until the desired number of samples are found and add a constraint that new generated instances should be far enough to instances that are found in the previous runs. The desired number of samples is set to 500 in this experiment. We also report the results of a random generation method, which randomly selects a feature, uniformly assigns a new feature value and weights it using the distance from the test instance at each generation of new instance, as used in LIME. In this experiment, since gradient-based methods are applicable only on numeric datasets, a process of conversion from nominal features to numeric features is conducted beforehead. The experimental results (Table 9) show that gradient based sampling is not as competitive as the proposed method, especially for high-dimensional data. One possible reason is that given the same size limit on the sample set, the instances generated by the gradient method are still not sufficiently diversified. Another major drawback of the gradient based method is that it is difficult to apply to datasets with nominal features.

6.6. Impact of explanation selection

In this part, we investigate the impact of explanation selection (the last step of PALEX) on the faithfulness of PALEX. The explanation selection process is a key step in the proposed method. The goal of this step is to reduce the number of explanation candidates and preserve the quality. The number of explanations K is set to 5. We compare the performance (best F_1 score) of the explanations *before* and *after* pruning in Table 10. It can be seen that the performance of the explanations before pruning is an upper bound for those after pruning, because the explanations of the latter are a subset of the former.

Dataset	Pattern-based	Gradient-based	Random
Balloon	**0.92**	0.88	1.00
Blood	**0.79**	0.69	0.72
Breast-cancer	**0.94**	0.72	0.68
Diabetes	**0.85**	0.42	0.73
Iris	**1.00**	0.56	0.85
Ionosphere	**0.63**	0.35	0.49
Labor	**0.96**	0.65	0.59
Musk	**0.86**	0.64	0.56
Titanic	**0.77**	0.26	0.65
Vote	**0.94**	0.79	0.65

Table 9. Faithfulness (F_1 score) of different sampling methods on numeric data

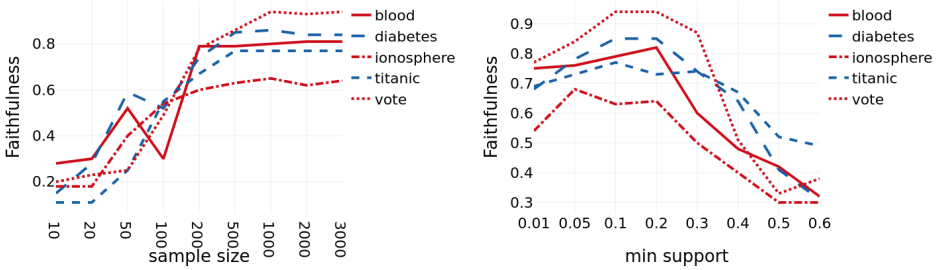
The numbers of explanations (#Expl) show that PALEX before pruning generates more than 50 explanations for datasets with more than four features, and the selection procedure consistently extracts less than four explanations, which suggests that the output remains the same for any choice of threshold K that is greater than 4. In terms of the quality, the explanations after pruning achieve comparable metrics in most datasets. The running time (in seconds) is also reported. The running environment is on Windows 10 with processor Intel(R) Core(TM) i7-5500U CPU 2.40GHz and 8.00 GB RAM.

	Precision		Recall		F_1		#Expl		Running time (s)	
Dataset	before	after	before	after	before	after	before	after	before	after
Balloon	**1.00**	**1.00**	**0.88**	**0.88**	**0.92**	**0.92**	3.52	1.0	0.81	3.13
Blood	**1.00**	0.97	**0.85**	0.72	**0.90**	0.79	6.3	2.4	0.15	1.33
Breast-cancer	**0.96**	**0.96**	**0.92**	**0.92**	**0.94**	**0.94**	157.3	2.0	0.17	0.28
Diabetes	**1.00**	0.98	**0.97**	0.82	**0.96**	0.85	76.1	2.4	0.64	1.03
Iris	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	8.0	3.1	0.08	0.22
Ionosphere	**0.93**	0.90	**0.86**	0.57	**0.81**	0.63	244.6	2.6	1.40	3.78
Labor	**1.00**	0.96	**1.00**	0.97	**1.00**	0.96	526.5	2.0	0.23	0.51
Musk	**0.91**	**0.91**	**0.83**	0.81	**0.88**	0.86	725.1	3.8	30.54	58.00
Titanic	**0.89**	0.78	**0.96**	0.82	**0.91**	0.77	53.4	2.3	0.43	0.72
Vote	**1.00**	0.99	**1.00**	0.92	**1.00**	0.94	164.1	1.6	0.20	0.34

Table 10. Impact of selection process. #Expl denotes the number of explanations, "before" denotes the metrics before the pruning process and "after" denotes the metrics after the pruning process.

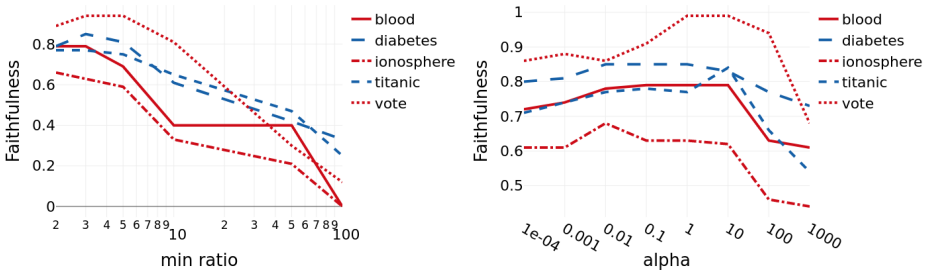
6.7. Impact of parameters

We also conducted experiments on a range of UCI datasets (blood, diabetes, ionosphere, titanic and vote) to investigate the impact of the parameter settings used by PALEX, and the results are shown in Figure 3. The performance (F_1 score) increases as the sample size N increases, then becomes stable after $N \geq 500$. A large sample size may slow down the running time as it has an impact on the later contrast pattern mining process (the higher the sample size is, the slower the mining process will be), and the experiments show that $N = 500$ is a reasonable setting as it is the minimum threshold that achieves good performance on average.The performance increases as $minSupp$ increases, and then it reaches a maximum and decreases after $minSupp$ is greater than a specific value (which is usually between 0.1 to 0.2, and depends on the specific dataset). PALEX achieves the best metrics on average when $minSup = 0.1$. The



(a) Impact of sample size (with $minSupp = 0.1$, $minRatio = 5$).

(b) Impact of $minSupp$ (with sample size $N = 500$, $minRatio = 5$).



(c) Impact of $minRatio$ (with $minSupp = 0.1$, $N = 500$)

(d) Impact of α (with $minSupp = 0.1$, $N = 500$, $minRatio = 5$)

Fig. 3. Impact of parameters on faithfulness (F_1 score)

performance decreases as $minRatio$ increases and achieves the best results on average when $minRatio = 2$. As in the case of high $minSupp$ and high $minRatio$ the number of patterns reduces substantially, hence it misses many valid explanations and causes degradation in the the performance as can be seen in Figure 3b and Figure 3c.

At last, we investigate the impact of α (regularization rate in the explanation selection process), and K is set to 5 (K is required to be larger than 1 to enable the influence of α). A small α would generate explanations that are similar to each other, thus misses some explanations of good quality, and a large α pushes the explanations to be distinct to each other while significantly ignoring their quality. The performance (best F_1 score of K explanations) is plotted in Figure 3d and it achieves best performance on average when $\alpha = 1$.

7. Limitations and Future Work

We have shown the effectiveness and usefulness of PALEX and compared it to the benchmark methods, and now we turn to its limitations and possible directions for future work.

One limitation is that PALEX assumes the availability of a set of frequent patterns that summarizes the possible local classification regions/subspaces of the training data.

If the pattern set is not of good quality (e.g., it cannot cover the whole sample space, or there is too much overlap), then the generation of neighbours will be degraded. It would be an interesting direction to push the frequent pattern mining phase within PALEX, to provide control over what type of frequent patterns are generated, with a view to optimisation of the downstream explanation task.

Another limitation is with respect to the evaluation challenges. The most common evaluation strategy is to evaluate the explanations extracted by the explanation extraction methods with a white-box model (from which the true model explanation can be inferred), and the model is treated as black-box for these methods. When evaluating with a real black-box model, the true explanations are never be available. One alternative might be to collect the true explanations from experts with background knowledge, but the issue is that there is often a gap between the model and the experts.

We also hope that local explanation methods encourage the collection of explanations while collecting the data. For example, the explanations could come from doctor reports in the medical diagnosis area, or from user reviews in a recommendation system. With such additional information, we may be able to improve a classifier to make more reliable predictions whilst the accuracy can be preserved.

8. Conclusion

In this work we have proposed PALEX, a pattern aided approach to provide explanations for individual predictions. PALEX exploits the intrinsic interpretability of contrast patterns and uses them as a form of explanation and generates the vicinity of the instance being predicted using frequent pattern spaces. An explanation selection process is also proposed to prune large candidate sets. Experimental results show that PALEX is more faithful to models compared to benchmark methods and the selection process is able to effectively choose K representative explanations.

References

Adler, P., Falk, C., Friedler, S. A., Rybeck, G., Scheidegger, C., Smith, B. and Venkatasubramanian, S. (2016), Auditing black-box models for indirect influence, *in* 'ICDM'16', IEEE, pp. 1–10.

Aggarwal, C. C. and Han, J. (2014), *Frequent Pattern Mining*, Springer.

Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K. and Müller, K.-R. (2010), 'How to explain individual classification decisions', *Journal of Machine Learning Research* **11**(Jun), 1803–1831.

Barakat, N. and Diederich, J. (2005), 'Eclectic rule-extraction from support vector machines', *International Journal of Computational Intelligence* **2**(1), 59–62.

Caruana, R., Lou, Y., Gehrke, J., Koch, P., Sturm, M. and Elhadad, N. (2015), Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission, *in* 'Proc. of KDD'15', ACM, pp. 1721–1730.

Craven, M. W. and Shavlik, J. W. (1996), 'Extracting tree-structured representations of trained networks', *Advances in Neural Information Processing Systems* pp. 24–30.

Dong, G. and Bailey, J. (2012), *Contrast data mining: concepts, algorithms, and applications*, CRC Press.

Duivesteijn, W. and Thaele, J. (2014), Understanding where your classifier does (not) work–the scape model class for emm, *in* 'ICDM'14', IEEE, pp. 809–814.

Fan, H. and Ramamohanarao, K. (2006), 'Fast discovery and the generalization of strong jumping emerging patterns for building compact and accurate classifiers', *IEEE Transactions on Knowledge and Data Engineering* **18**(6), 721–737.

Freitas, A. A. (2014), 'Comprehensible classification models: a position paper', *ACM SIGKDD Explorations Newsletter* **15**(1), 1–10.

Fung, G., Sandilya, S. and Rao, R. B. (2005), Rule extraction from linear support vector machines, *in* 'Proc. of KDD'05', ACM, pp. 32–40.

Goodman, B. and Flaxman, S. (2016), Eu regulations on algorithmic decision-making and a "right to explanation", *in* 'ICML Workshop on Human Interpretability in Machine Learning (WHI 2016)'.

Han, J., Pei, J. and Yin, Y. (2000), Mining frequent patterns without candidate generation, *in* 'ACM sigmod record', Vol. 29, ACM, pp. 1–12.

Henelius, A., Puolamäki, K., Boström, H., Asker, L. and Papapetrou, P. (2014), 'A peek into the black box: exploring classifiers by randomization', *Data Mining and Knowledge Discovery* **28**(5-6), 1503–1529.

Henelius, A., Puolamäki, K., Karlsson, I., Zhao, J., Asker, L., Boström, H. and Papapetrou, P. (2015), Goldeneye++: A closer look into the black box, *in* 'International Symposium on Statistical Learning and Data Sciences', Springer, pp. 96–105.

Jia, Y., Bailey, J., Kotagiri, R. and Leckie, C. (2018), 'Pattern-based feature generation', *Feature Engineering for Machine Learning and Data Analytics* p. 245.

Kang, S. and Ramamohanarao, K. (2014), A robust classifier for imbalanced datasets, *in* 'PAKDD', pp. 212–223.

Karmarkar, N. (1984), A new polynomial-time algorithm for linear programming, *in* 'Proceedings of the sixteenth annual ACM symposium on Theory of computing', ACM, pp. 302–311.

Koh, P. W. and Liang, P. (2017), 'Understanding black-box predictions via influence functions', *arXiv preprint arXiv:1703.04730* .

Kohavi, R. (1995), 'The power of decision tables', *Machine learning: ECML-95* pp. 174–189.

Kurfess, F. J. (2000), 'Neural networks and structured knowledge: Rule extraction and applications', *Applied Intelligence* **12**(1), 7–13.
URL: *http://dx.doi.org/10.1023/A:1008344602888*

Letham, B., Rudin, C., McCormick, T. H., Madigan, D. et al. (2015), 'Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model', *The Annals of Applied Statistics* **9**(3), 1350–1371.

Martens, D., Baesens, B., Van Gestel, T. and Vanthienen, J. (2007), 'Comprehensible credit scoring models using rule extraction from support vector machines', *European Journal of Operational Research* **183**(3), 1466–1476.

Martens, D. and Provost, F. (2011), 'Explaining documents' classifications', *Center for Digital Economy Research* .

Ribeiro, M. T., Singh, S. and Guestrin, C. (2016*a*), Model-agnostic interpretability of machine learning, *in* 'ICML Workshop on Human Interpretability in Machine Learning (WHI 2016)'.

Ribeiro, M. T., Singh, S. and Guestrin, C. (2016*b*), Nothing else matters: Model-agnostic explanations by identifying prediction invariance, *in* 'NIPS Workshop on Interpretable Machine Learning in Complex Systems'.

Ribeiro, M. T., Singh, S. and Guestrin, C. (2016*c*), Why should i trust you?: Explaining the predictions of any classifier, *in* 'Proc. of KDD'16', ACM, pp. 1135–1144.

Ribeiro, M. T., Singh, S. and Guestrin, C. (2018), Anchors: High-precision model-agnostic explanations, *in* 'AAAI Conference on Artificial Intelligence'.

Robnik-Šikonja, M. and Kononenko, I. (2008), 'Explaining classifications for individual instances', *IEEE Transactions on Knowledge and Data Engineering* **20**(5), 589–600.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D. and Batra, D. (2017), Grad-cam: Visual explanations from deep networks via gradient-based localization., *in* 'ICCV', pp. 618–626.

Shang, J., Tong, W., Peng, J. and Han, J. (2016), Dpclass: An effective but concise discriminative patterns-based classification framework, *in* 'Proc. of SDM'16', SIAM, pp. 567–575.

Wang, F. and Rudin, C. (2015), Falling rule lists., *in* 'AISTATS'.

Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M. and De Freitas, N. (2016), Dueling network architectures for deep reinforcement learning, *in* 'Proc. of ICML'16', JMLR.org, pp. 1995–2003.
URL: *http://dl.acm.org/citation.cfm?id=3045390.3045601*