Trade-offs in Explanatory Model Learning

Madalina Fiterau
Carnegie Mellon University
mfiterau@cs.cmu.edu

March 21, 2012

Abstract

In many practical applications, accuracy of a prediction is as important as understandability of the process that leads to it. Explanatory learning emerges as an important capability of systems designed for close interaction with human users. Many generic white-box predictive model types are readily available and potentially appropriate for the task (decision trees, association rules, sub-spacing, etc.) with more being developed each day. This project introduces an algorithm specifically designed to learn explainable models: Explanation Oriented Partitioning (EOP). Its performance is compared against a range of relevant alternatives using multiple synthetic and real-world data sets. EOP typically yields substantially simpler, more interpretable models, while maintaining comparable predictive accuracy.

1 Introduction

Typical design methodology of data-driven analytic systems focuses on optimization of an empirical loss function appropriate for the task. In predictive scenarios, such as classification or regression, considerable efforts are often spent on model selection and tuning so that quantitative metrics of accuracy and reliability of the solution are optimized. Similarly, in descriptive analytics we often focus on maximizing the fidelity with which the underlying data mechanisms are represented, while minimizing the risk of overfitting. The emphasis on accuracy of the resulting models often trumps desire for explainability. This can be seen among popular out-of-the-box high performers such as boosting or random forests, which often appear to the user as black-box oracles with very little to offer in terms of explaining their predictions. We acknowledge the practical need for such methods. We observe that in many field deployment scenarios, especially in the context of data mining, the end users are ready to trade off some loss of accuracy for ease of understanding of the results. These users require explainable, white-box models, which would maintain reasonable accuracies.

We introduce Explanation-Oriented Partitioning (EOP), a method that is designed for this task. It uses a few low-dimensional projections of data, each with its own discriminator, to learn explainable classifications. This meta-algorithm can work with discriminators of various types, such as SVMs, logistic regression, or with the non-parametric k-nearest neighbors. It leverages local performance of the classifier to identify low-dimensional regions (easy to present and interpret) of the feature space where data is well-classifiable. EOP picks out multiple such clusters, maximizing expressiveness while maintaining compactness of the resulting model. The individual result for a test data point includes concise information about the region of feature space which clearly supports the current prediction, in addition to the label of the most likely class. As a useful side effect, EOP can also identify regions of the feature space where data is particularly noisy and difficult to accurately discriminate.

EOP is an iterative algorithm. In the first iteration, it selects the one projection that is most effective in support of the classification task, among all data projections of a given dimensionality. The data that cannot be accurately classified and explained using the current model becomes the focus of the next iteration. This way we obtain a hierarchical sequence of models similar to a decision list, each component of which is a projection of data that can be used to classify and explain the partition assigned to it accurately and without excessive complexity.

The hierarchical flavor of EOP makes it somewhat similar to boosting [11]. The way it splits the data brings up a reference to partitioning models such as CART [4]. Empirical evaluation shows that, compared to

boosting, EOP produces white-box and often more compact models at the price of a slight loss in classification accuracy. Compared to CART, EOP models tend to be more accurate at low complexities, and they require fewer projections of data to provide superior explainability.

EOP is also an ensemble model. Ensemble learning has been long known to enable great improvements of model accuracy by combining the capabilities of multiple base classifiers [25, 3, 1, 20]. Methods such as Winnow [14] and Boosting [10] are guaranteed to decrease training error with each iteration by tweaking either the voting coefficients or the weights of the training data records. Performance can be futher enhanced by combining those techniques [23]. However, there has been much well justified debate as to what extent the accuracy of prediction is indicative of the ability of an algorithm to uncover the processes behind data [2]. To answer that, approaches that simplify trained ensembles have been proposed [8, 7], and methods that replace accurate black-box models with more interpretable equivalents [6, 15]. Other techniques attempt to improve understandability by simplifying or compressing the feature space [13, 24, 19]. So far only a handful of algorithms have been specifically designed to yield understandable models. However, rules learned as in [17] can be hard to visualize, and itemset mining [16] is not quite native for classification tasks. In Feating [22], submodel selection relies on simple attribute splits followed by fitting local predictors. EOP reverses this sequence: it first tests - which are more generic than those found by Feating - to identify useful discriminators, and then it makes splitting decisions based on their performance.

We empirically compare EOP to a representative group of the methods listed above. Our experiments show how EOP finds succinct descriptions that capture patterns in data. This ability is a crucial aspect of practical utility of prediction systems working in close interaction with human users.

2 Explanation-Oriented Partitioning

2.1 EOP Learning

The Explanation-Oriented Partitioning algorithm iteratively selects projections of data in which the data can be classified with high accuracy. Here it can use any externally supported classifier - in the experiments shown below we use Support Vector Machines [5] and decision stumps. In each of the selected projections, EOP identifies contiguous areas (regions) in which predictions are consistently accurate. These regions are then used in lieu of explanations for predictions made for data inside their bounds. An example prediction produced by the trained EOP model in response to a test data query could, for instance, pronounce: *'This query appears to belong to class A. It can be shown that in the scatterplot of data projected onto x_1 and x_7 this query is densely surrounded by instances that belong to the same class in the area bounded by* $13.5 < x_1 < 25.0$ *and* $0.4 < x_7 < 1.3$*'*. The user therefore obtains the context of the prediction and the ability to visually confirm its sensibility.

The algorithm employs a few parameters. The users can specify the target classification error rate ϵ, the regularization parameter for the used classifiers λ - its meaning is obviously specific to the chosen classifier type, and η used to control complexity of the projections of data. The users supply the training data and the algorithm exhaustively evaluates all feasible projections of a selected dimensionality - we use 2-dimensional projections in the experiments considered below. EOP then identifies the projection π which allows for the most accurate classification of data given the particular settings of parameters (λ, η), and the corresponding trained classifier h. The next step is to identify regions in the current projection where the data is predominantly correctly classified, with the maximum within-region classification error rate of ϵ. There may be multiple such potentially overlapping regions in any of the considered projections. EOP uses a distinct validation subset of data to calibrate identified regions by expanding or contracting their boundaries, or even deleting some of them, to prevent overfitting. Finally, the training data captured by the calibrated regions is removed from consideration, and the remainder becomes the input for the next EOP iteration. Algorithm 1 presents the pseudo-code for learning an EOP model from data.

The resulting model is therefore a hierarchy of projections of data, corresponding trained classifiers and regions selected in these projections. When the EOP model is queried with a test data point, the top component of the hierarchy is inspected first. If the query falls inside any of the regions associated with this sub-model, its classifier predicts the class label, and it is returned to the user together with the description of the invoked region. Otherwise, the algorithm falls back to the next component of the hierarchy. Therefore,

the query-time operation is that of a decision list.

Algorithm 1 EOP Algorithm

```
EOP(Data, ϵ, λ, η)
(trainingData, calibrationData) = Split(Data)
Classifiers = []
Regions = []
while TrainingData is not empty do
  (h, π) = SelectClassifier(trainingData, λ, η)
  trainError =
  (ApplyClassifier(trainingData.π) ≠ currentData.output)
  calibrationError =
  (ApplyClassifier(calibrationData.π) ≠ calibrationData.output)
  sets = ObtainSets(trainingData,trainError,ϵ)
  FilterSets(sets,calibrationData,calibrationError)
  if pointsInRegion is Empty then
    increase(ϵ)
  else
    Classifiers.append(h)
    Regions.append(sets)
    pointsInRegion = PointsInRegion(trainingData,sets)
    currentData.eliminate(pointsInRegion)
  end if
end while
return (Regions,Classifiers)
SelectClassifier(data, λ, η)
Π = CombineFeatures(data)
for all π ∈ Π do
  h.append(TrainClassifier(π,λ))
  rediction = ApplyClassifier(h,π)
  error = mean(prediction not equal data.output)
  score.append(error + size(π)*η)
end for
idxBest = index(score,min[score])
return (h[idxBest],Π[idxBest])
```

The basic stopping criterion for EOP's learning procedure is the exhaustion of training data. It may not be attainable if the required accuracy of classification in a region, ϵ, is overly restrictive. If so, EOP can either dynamically relax ϵ until all training data is accounted for by the model, or it can leave a certain amount of hard to handle data unresolved. The choice depends on the requirements of the application.

2.2 Implementations of EOP Region Finding

So far we have described the high-level EOP algorithm without providing specifics regarding region extraction. We have mentioned that EOP is flexible regarding the choice of the base classifier. It can also rely on various approaches of region extraction. Below we detail two such methods, a parametric and a non-parametric segmentation of data, outlining the trade-offs that come with each of them and how bootstrapping can help in making the process robust.

2.2.1 Bounding Polyhedra

One way of characterizing regions of consistently classifiable data is to encase them in simple boundaries, such as polyhedra. In order for a polyhedron to qualify as a region of interest, the fraction of misclassified

data in it should not exceed ϵ. Given a particular projection of data, the task of finding a polyhedron that maximizes data coverage while satisfying the ϵ condition is NP-hard. Instead, simple heuristics can be used to achieve satisfactory results.

We used a method which starts with a randomly selected correctly classified data point as a seed, and subsequently adds more such points located nearby, growing the region for as long as the minimum accuracy constraint can be maintained. After such a set is found, the process is restarted with another correctly classified data point that is not yet enclosed in any of the previously constructed polyhedra.

The process ends when all the correctly classified data is consumed. The algorithm allows the resulting polyhedra to overlap. In order to prevent overfitting, calibrate the result using a hold-out subset of data. Polyhedra that do not include any of the calibration data or for which the mean error over the enclosed calibration data exceeds ϵ are deemed unreliable and removed. The remaining regions are subject to shape adjustments to better represent the calibration data.

The EOP learning algorithm takes into account the complexity of the set of polyhedra that survive calibration. We estimate complexity as the sum of the number of facets of each polyhedron across all polyhedra associated with the particular classifier. Note that the presented process can be easily tailored to search for simplexes or hyper-rectangles. The latter are especially attractive from the potential end-user perspective if we require their sides to align with the axes of the data coordinate system. The resulting region boundaries can then be expressed using highly intuitive interval queries. Also, geometric boundaries of regions do not have to be linear. Elliptical bounds can be used as well.

2.2.2 Nonparametric Regions

Presenting patterns in a parametric form is, although intuitive, not the only way to express conditions imposed on the involved data points. An alternative is to estimate the density of the correctly/incorrectly classified data and define the region using a threshold on the likelihood ratio, so that the data with likelihood ratios greater than a learned threshold would be considered easily classifiable and therefore eligible for inclusion in one of the reported regions.

A potential problem is that the regions found in this manner are highly dependent on the properties of a chosen density estimation method and its parameters such as e.g. bandwidth, selection of which is a design problem on its own. We go around this issue by not estimating the densities of correctly and incorrectly classified data. Instead, we score each candidate data point using the information of the distances to its correctly and incorrectly classified neighbors. The number of neighbors considered is $k = \frac{1}{\epsilon}$. The intuition is that if the point being scored is a part of a contiguous region that satisfies the required accuracy, it should not have more than one incorrectly classified data within its k-neighborhood. This property can be used for pruning data unfit for inclusion in any of the regions worth reporting, and bounding the search for computation time savings.

We compute a weight for each of k neighbors of point p: its i^{th} neighbor n_i is assigned a weight of $w_i = \frac{1}{1+d(p,n_i)}$, where $d(p, n_i)$ denotes the distance between p and n_i. The score is then computed as the ratio of the sum of weights of the correctly classified neighbors to the sum of weights of all k neighbors:

$$Score(p) = \frac{\sum_{i=1}^{k} \frac{1}{1+d(p,n_i)} C(n_i)}{\sum_{i=1}^{k} \frac{1}{1+d(p,n_i)}}$$

$C(n_i) = 1$ if n_i is correctly classified, 0 otherwise.

We compute the scores for the complete set of training data. The next step is to determine the threshold of the score to decide which of the correctly classified data points should be included in a region to be reported. We identify a subset S_g of data with scores greater than $1 - \epsilon$. The region eligibility threshold is then set as the lower of $1 - \epsilon$ and $\frac{|S_g \cap S_c|}{|S_g|}$ where S_c is the subset of correctly classified training data. Similarly to the parametric approach, we can use calibration data to adjust the eligibility threshold in order to robustify the learned nonparametric EOP regions against overfitting.

2.3 EOP Operation

2.3.1 Example using Synthetic Data

Let us consider a simple example to illustrate EOP operation. We synthesized a data set with 3-dimensional continuous input space and a binary output. The data belonging to the first class (depicted in red in the graphs below) follow a uniform distribution [0,5] over all 3 features. The points in the second class (depicted in blue) have been generated using two models each composed of a bi-variate Gaussian and a uni-variate uniform distribution. One of the models was Gaussian w.r.t. features 1 and 2, and uniform in dimension 3, while the other was Gaussian for features 1 and 3, and uniform in feature 2. The Gaussians used in data generation had the following parameters:

$$\mathcal{N}_{12} \sim \left(\mu = \begin{pmatrix} 0.3 & 0.5 \end{pmatrix}^T, \Sigma = \begin{pmatrix} 1.5 & 0.7 \\ 0.7 & 1.5 \end{pmatrix}\right)$$

$$\mathcal{N}_{13} \sim \left(\mu = \begin{pmatrix} 0.7 & 0.3 \end{pmatrix}^T, \Sigma = \begin{pmatrix} 1.2 & 0.3 \\ 0.3 & 1.2 \end{pmatrix}\right)$$

. The class priors are uniform. The Gaussian patterns of the second class generate equal number of data points. Figure 1 shows this data projected on all combinations of pairs of features.

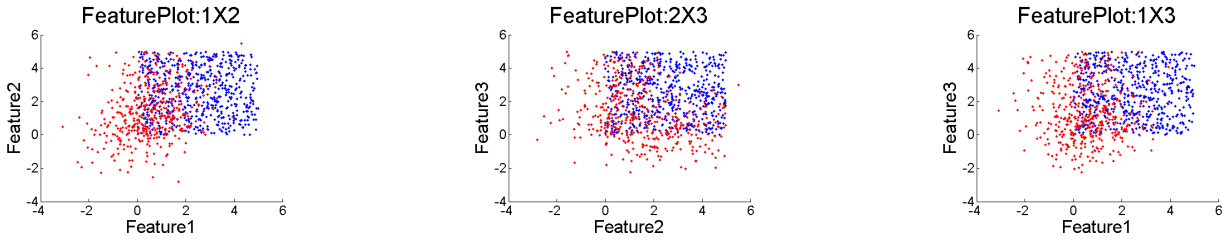


Figure 1: 2-D projections of synthetic data

Each row of graphs in Figure 2 illustrates one iteration of the EOP algorithm. For each row, the graph on the left represents the scatterplot of data considered at that iteration - points belonging to different classes are shown with distinct symbols: '+' for label 1 and 'o' for label 0. The center graph represents the probability of accurate classification computed for each data point computed using the k-nn score, shown as a colormap; the red side of the spectrum denoting the points on which the classifier will do well, while the blue end shows points where correct classification is unlikely. Finally, the graph on the right represents the assignment of data to the reportable regions, the '+' symbol marking data that is deemed to belong to the regions.

The first iteration of EOP selects the projection of data onto features 1 and 3 as enabling the most accurate classification overall. The top left graph in Figure 2 shows the training data in this projection with the color and symbol-coded class labels. A classifier h_1 is trained on this 2-D problem. The data that h_1 classifies correctly are marked with '+' in the central graph, while the misclassified data points are marked with 'o'. The color intensity of these symbols notifies the proximity to neighboring correctly classified training examples. As expected, the classification is more confident at the farther sides of the classification boundary, and not so convincing wherever the training data shows significant overlap of the two classes. The top right graph depicts data considered sufficiently explainable by the above described nonparametric procedure to be included in region R_1 with '+'. In the prediction phase, any data point belonging to R_1 will be classified using h_1.

To continue the iterative process, the training data put in R_1 are eliminated from consideration in subsequent iterations. In the second iteration, the projection on features 1 and 2 is picked with a corresponding linear separator h_2, as shown in the second row of graphs in Figure 2. Again, the points that are the easiest to classify are the ones located in the areas with little class overlap. They will be removed from consideration in the subsequent steps of the procedure.

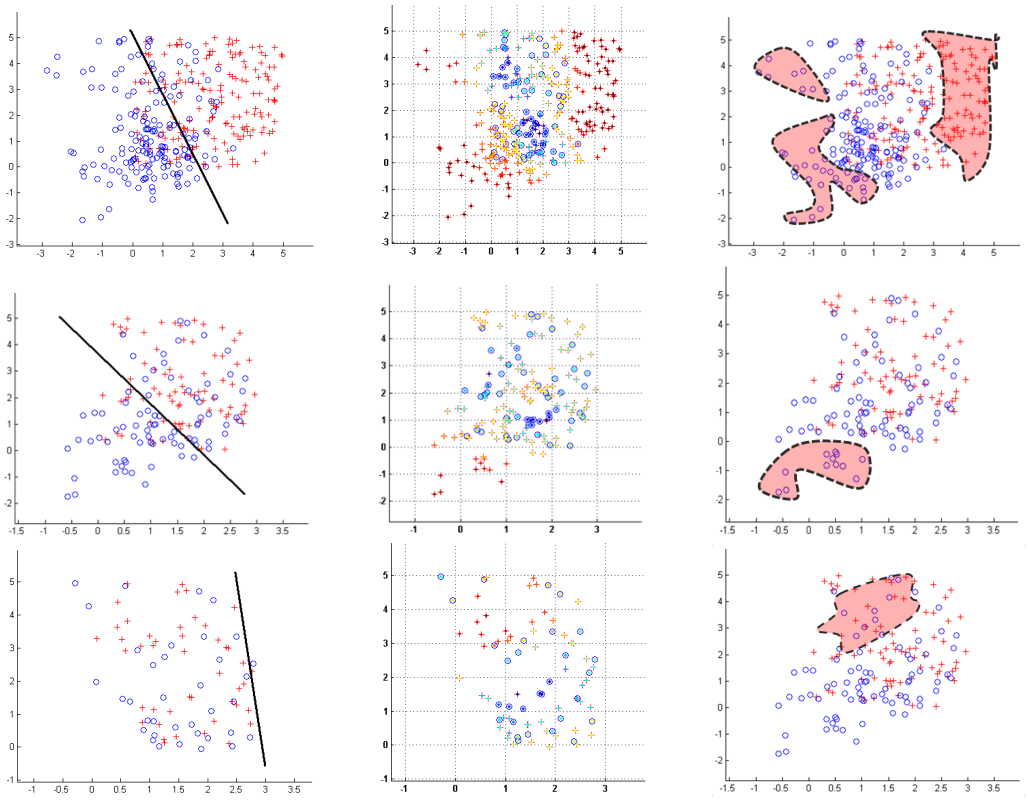


Figure 2: The first 3 iterations of nonparametric EOP executed on synthetic data set - one row per iteration.

In the third iteration, the remaining data is projected on features [1,3] again. The model carves out a region of consistent classification located in the top left of the diagram. If we let the process to continue it would have terminated after 5 iterations when all training data points would have been expended.

When a new data point is to be classified, it is first projected on features [1,3]. If it belongs to R_1, it is classified with h_1, otherwise it is projected on [1,2] and classified with h_2 if it belongs to R_2. Otherwise it is passed on to the following projection and so on. If the point cannot fit any of the learned regions, it can be either left unclassified or assigned the label of the most common class - we use the latter in the experiments. To illustrate how the parametric model works with the same data, EOP using rectangular regions was executed. Figure 3 shows all of the selected rectangular regions for $\epsilon = 0.1$ (left) and the regions that survive pruning with validation data (right). This example illustrates importance of using validation data in preventing explosion of complexity and overfitting.

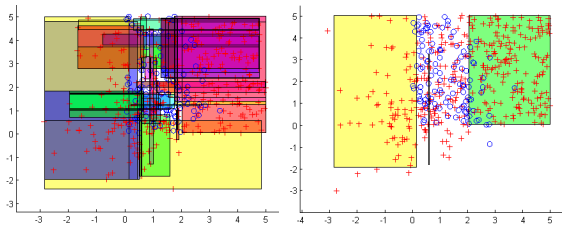


Figure 3: Selection of consistently classifiable regions and results of pruning shown using the synthetic data.

2.3.2 Avoiding unnecesary complexity

Let us consider a variation of the classic XOR problem: a two-dimensional binary data consisting of four separable regions symmetrically distributed as shown in the top left plot of Figure 4. Data belonging to class 1 (shown in red) occupies two square regions shifted diagonally apart. Class 0 data (shown in blue) covers two triangular regions filling in the cavities left by the class 1 distribution. Decisions on both features are

required to correctly separate the two classes. Decision trees are known to have difficulties with such data, mostly because their learning algorithms follow greedy strategies of maximizing immediate gains at each step of tree development. Therefore, in our example, they end up chunking the data into many small slices, instead of discovering a visually obvious geometric pattern of data distribution. Consistent with the typical behaviour of decision trees, the top ten decisions in the learned CART tree explain the data in a roundabout manner, as depicted in the top right picture in Figure 4. A regularized model obtained with CART consists of 21 nodes - substantially more complex than the theoretically optimal model with only 3 nodes.

EOP, albeit not perfect, fares substantially better with regard to complexity in this example. It starts by training the best linear separator of the complete distribution of data, which, unsurprisingly, is equivalent to a default classifier and it classifies all data as belonging to the more populous class 1. The training samples of class 1 are then marked as correctly classified, and two regions of highly reliable classification are identified as shown in the bottom left graph of Figure 4. In the next iteration, EOP identifies regions covered by the points of class 0 as shown in the bottom right of Figure 4. The hierarchy of the resulting model will have 3 levels: one for the regions of class 1, one for the regions of class 0, and the third level will collect data left over from previous iterations. These data points are located close to the boundary between the two classes and EOP could not confidently place them in any of the regions, therefore it will either refuse to commit to classifying them, or label them as members of the most frequent class.

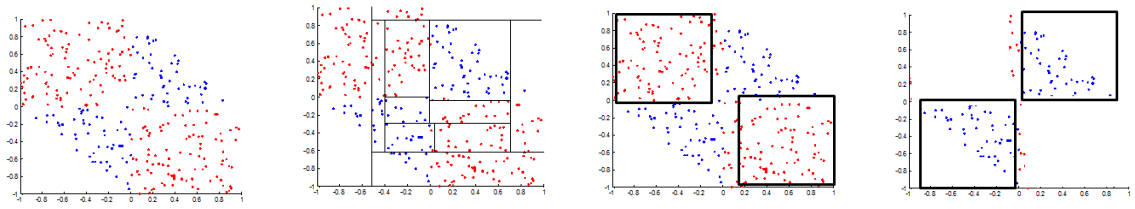


Figure 4: XOR data (left). Corresponding decisions learned by CART (center-left). First set of regions found by EOP (center-right). Second set of regions found by EOP (right)

3 Experimental Evaluation

The previous examples provide intuition for how EOP can find concise patterns in data. Now we focus on quantitatively comparing the performance of EOP versus classical alternatives, AdaBoost and CART, as well as a handful of more contemporary algorithms (random forests [21], multiboosting [23], subspacing [13], and feating [22]), using realistically complex synthetic and real-world data. The results indicate that EOP is comparable in terms of the attainable classification accuracy to these alternatives. However, EOP achieves it using simpler models and more informative initial projections with easily classifiable subsets of data.

We used synthetic data and multiple real world datasets drawn from the UCI repository [9]. The two-class synthetic data was generated by sampling from uniform distributions along all coordinates. Then, we injected additional data drawn from randomly shaped and positioned Gaussians spanning a number (lower than the overall dimensionality of data) of randomly selected dimensions. Each such Gaussian produced data of only one of the two classes, with the class label drawn uniformly. Each synthetic set included 7 such injections: one 4-dimensional, two 3-dimensional and four 2-dimensional. The data generated in this manner can be easy to classify if the injected patterns are populous and sufficiently disjoint, or more difficult to classify if the patterns overlap or involve a small number of instances. We created 10 datasets of varying difficulty, each with 10 real-valued input features, one binary class output, and 3,000 data points. In UCI datasets, we ignored any non-numeric input features, if present. We used *Breast Cancer Wisconsin* (10 inputs, 569 records), *MiniBoone* (10 inputs, 5000 records), *Breast Tissue* (9 inputs, 1696 records), and *Vowel* (10 inputs, 990 records) data. Two thirds of each were available for training models, the rest was used for testing, and only the test set results are presented below.

We also successfully executed EOP on larger datasets (up to 10,000s of features and 100,000s of records

so far) replacing the standard exhaustive approaches to projection selection and region identification with randomized sampling.

3.1 Accuracy and Complexity

3.1.1 Comparison to Boosting

Adaboost trains a sequence of weak classifiers by increasing at each iteration the weight of the points that were incorrectly classified at previous iterations. The training set error decreases at each step, and is theoretically guaranteed to go to zero in the limit, after a sufficient number of iterations. In practice, testing set error often reaches a non-zero, albeit small, plateau.

EOP is similar to Adaboost in its iterative subsetting of data into increasingly difficult subproblems. It also follows the basic principle: no matter how fundamentally incompetent a particular classifier is, it will often do well in some part of the feature space. However, the intents differ: boosting primarily tries to lower error rates, while EOP prioritizes explainability of the models, while trying to maintain decent accuracy.

Table 1: Comparison of nonparametric EOP (E) and boosting (B) - both with SVM base classifiers - in terms of accuracy (A) and complexity (C) on artificial data

	B A	***B C***	***E A***	***E C***
DS1	0.97	48	0.964	22.53
DS2	0.904	63	0.903	25.5
DS3	0.97	217	0.964	39.12
DS4	0.928	39	0.922	28.21
DS5	0.944	97	0.928	28.97
DS6	0.918	149	0.931	59.87
DS7	0.954	206	0.964	27.63
DS8	0.968	214	0.967	23.08
DS9	0.978	9	0.976	27.67
DS10	0.914	138	0.895	41.45
Mean	0.9448	118	0.941	32.403
Stdev	0.027	77.896	0.029	11.489
T-test	0.832	0.003		

Table 1 presents the outcome of running nonparametric EOP and Adaboost on synthetic data. The classification error is low in both cases, and although boosting beats EOP on average by a small margin, the difference is not significantly systematic as indicated by the p-value of the paired T-test. On the other hand, EOP outperforms boosting in simplicity and the difference is significantly systematic. We use operational complexity as the metric of reference - the expected number of operations to be performed when a test data point needs to be classified. For boosting, this means a vector multiplication for each classifier, while in the case of nonparametric EOP it is the expected number of neighbors that need to be taken into account before a decision is reached plus the classification effort - one vector multiplication.

3.1.2 Comparison to CART

A prototypical white box method, CART, learns decision trees for classification by splitting the feature space into regions that have consistent values of the output labels. Heavy pruning and cross-validation are used to prevent overfitting. The outcome of the algorithm is not just accurate, but also a meaningful model revealing some structural information about data. Additionally, because the feature space is split only as necessary, the resulting model can be compact. The trained CART decision tree classifies implicitly: once a leaf node is reached, the prevalent class label is chosen to answer the query.

Table 2: Comparison of nonparametric EOP - with decision stumps - (E) and CART (C) in terms of accuracy (A) and complexity (C) on artificial data

	CART A	***CART C***	***EOP A***	***EOP C***
DS1	0.947	21	0.924	16
DS2	0.918	15	0.903	7
DS3	0.951	27	0.930	8
DS4	0.838	41	0.754	5
DS5	0.927	31	0.887	9
DS6	0.873	33	0.830	6
DS7	0.880	27	0.828	15
DS8	0.957	27	0.923	24
DS9	0.934	33	0.891	7
DS10	0.959	17	0.936	11
Mean	0.918	27.2	0.881	10.8
Stdev	0.041	7.91	0.059	5.92
T-test	0.00012	0.00063		

EOP groups and filters data by how well a classifier can deal with them, so the assignment of the output class is indirect - the label is in fact assigned by the corresponding classifier. Also, EOP splits data differently than CART. Its sequential approach leads to a hierarchy structured as a list rather than a tree, often yielding lower complexity; nonetheless it is subject to similar dangers of overfitting. Since both methods produce human understandable models, a white-box comparison can be drawn. For fairness

to CART, we compare EOP models that use decision stumps as base classifiers (we could use fancier base classifiers to obtain higher accuracies).

Table 2 shows how nonparametric EOP fares against CART on synthetic data. The compared models - including the η and ϵ parameters of EOP - are obtained through cross-validation. While CART is on average about 3 percentage points more accurate, EOP uses models that are considerably less complex in terms of the number of weighted decisions (the weights are equal to the number of data dimensions used by decisions). The differences in performance and model complexity are significant in terms of paired T-test.

Table 3: Comparison of parametric EOP - with decision stumps - and CART in terms of accuracy (A) and complexity (C) on artificial data

	CART A	***CART C***	***EOP A***	***EOP C***
DS1	0.850	11	0.837	2
DS2	0.820	9	0.747	4
DS3	0.826	17	0.741	3
DS4	0.914	9	0.790	6
DS5	0.842	11	0.838	4
DS6	0.884	5	0.886	3
DS7	0.874	7	0.747	2
DS8	0.834	5	0.753	3
DS9	0.840	9	0.705	5
DS10	0.812	25	0.693	3
Mean	0.850	10.8	0.773	3.3
Stdev	0.031	6.07	0.062	1.49
T-test	0.00073	0.00197		

Table 3 summarizes performance of the parametric version of EOP as compared to CART. Although the accuracy is on average not quite as good as that of nonparametric EOP, there still are some datasets for which this model performs better than CART. Importantly, the parametric models are considerably less complex, and rely on easy to interpret regions. We present results for a parametric EOP that uses axis-aligned rectangular regions. Assignment of a testing data point to the appropriate region can be done using two vector comparison operations per tried region, so parametric EOP's complexity is proportional to the expected number of rectangles against which a query needs to be tested. Note that the results for CART differ between these tables due to randomness of the data generation process.

3.1.3 Real-world Data Evaluations

Qualitatively similar results were obtained when EOP and CART were compared using real-world datasets taken from the UCI repository. But for the sake of completeness of comparison, we added a few additional contemporary and relevant algorithms to our evaluations. Random Forest [21] is a popular and powerful black-box technique that learns a bagged ensemble of decision trees, each on a different bootstrap sample of the training data, in hopes to reduce the variance component of the predictive error. We find Random Forests highly competitive in many applications encountered in our practice. Random Subpsacing [13] learns a random forest by sampling a subset of features to train each tree. It is similar to EOP in how it tries multiple projections of data onto reduced-dimensionality subspaces, and how it allows the use of various splitting functions. Multiboosting [23] aims to bridge the gap between ensemble learning methods designed to reduce the bias component of the predictive error (e.g. boosting) with those that take on variance (e.g. bagging). In that, it is complementary to the other methods selected for our evaluations, as well as to EOP. Feating (feature-subspace aggregation) [22] is a relatively recent method that splits the data space through a decision tree and trains local models. It is similar to EOP in that the decision structures rely on discriminators tied the leaves. Feating submodel selection relies on simple attribute splits, followed by fitting local predictors. EOP reverses this sequence: it first uses tests - which more general than the ones in Feating models - to identify useful discriminators and

Table 4: Comparison of accuracy and model complexity obtained by different methods - Random Forests (RF), Multiboosting (Mb), Subspacing (Ss), Feating (FT), CART, nonparametric EOP (N-EOP) and parametric EOP (R-EOP) - on datasets from the UCI repository - Breast Cancer Winsconsin, MINIBoone, Breast Tissue, Vowel.

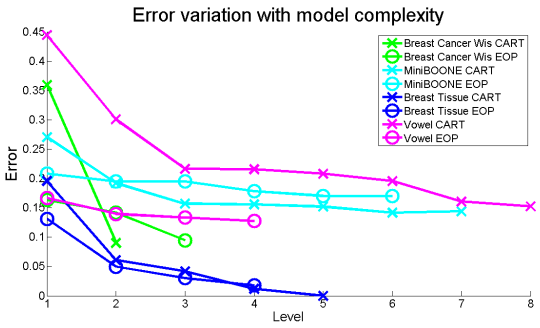
Acc	**RF**	**Mb**	**Ss**	**FT**	**CART**	**N-EOP**	**R-EOP**
BCW	0.942	0.922	0.912	0.938	0.908	0.905	0.894
MB	0.86	0.849	0.87	0.728	0.856	0.83	0.83
BT	1	0.943	1	0.956	1	0.982	0.78
Vow	0.944	0.841	0.899	0.868	0.947	0.872	0.842
Compl	**RF**	**Mb**	**Ss**	**FT**	**CART**	**N-EOP**	**R-EOP**
BCW	325	15	30	20	3	3	2
MB	2456	60	30	20	19	8	7
BT	18	15	30	20	15	4	2
Vow	516	60	30	20	31	8	4

them, based on their performance, it determines the span on which each discriminator will be active.

Table 4 summarizes the comparison. Although nonparametric EOP does not come first on accuracy, it typically outperforms one or two counterparts. However, in most cases it offers a substantial reduction in complexity - only CART matches nonparametric EOP on Breast Cancer data. Parametric EOP allows further savings of complexity at the expense of slight reduction of accuracy.

Figure 5 compares accuracy of nonparametric EOP and CART computed during learning the structure, at subsequent levels of the respective hierarchies. EOP achieves better performance for all datasets at the first level of hierarchy and for most of them at the second level. CART requires deeper structures to finally take the lead at the cost of additional complexity.

Figure 5: Variation of error with model complexity for CART(+) and nonparametric EOP(o) on datasets from the UCI repository



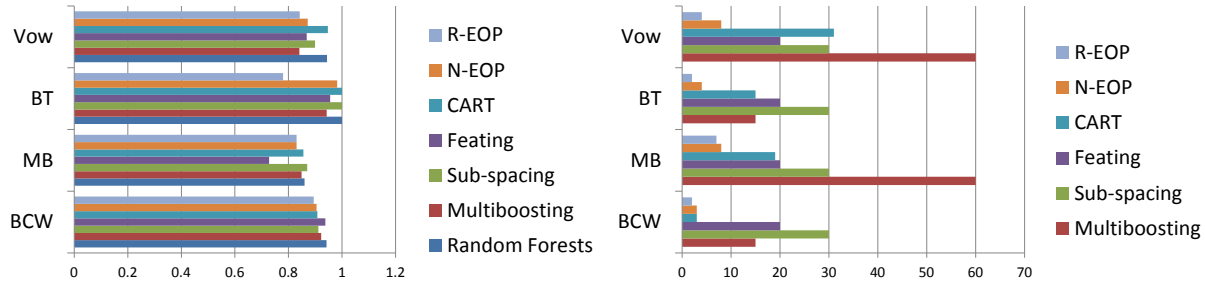


Figure 6: Accuracy (left) and complexity (right) of multiple methods on the UCI datasets.

Figure 6 shows how the considered white-box and black-box methods perform, in terms of accuracy and complexity, on UCI datasets. The graphs show some differences in terms of accuracy - more specifically, Random Forests is the best out of the black box models, while CART and sub-spacing are among the competitive white-box models. Overall, in terms of accuracy, there are few notable differences - Feating performs worse on the MiniBoone dataset and Rectanglar EOP. However, there certainly are differences as far as the simplicity of the models is concerned: the EOP models are consistently less complex than the rest.

Table 5: Complexity and accuracy of several methods on datasets from real applications

Accuracy	**RandF**	**Mboost**	**Sub-spacing**	**Feating**	**CART**	**N-EOP**	**R-EOP**	**Adaboost**
Mimic II	0.9935	0.9936	0.9936	0.9936	0.9933	0.9926	0.9941	0.9936
Cell Data	0.7811	0.7877	0.788	0.7877	0.7884	0.7311	0.7909	0.7877
Fuel	0.7095	0.6855	0.7174	0.7115	0.7107	0.554	0.5282	0.7033
Spambase	0.9143	0.7511	0.8997	0.8125	0.8813	0.8461	0.8304	0.8615
Complexity	**RandF**	**Mboost**	**Sub-spacing**	**Feating**	**CART**	**N-EOP**	**R-EOP**	**Adaboost**
Mimic II	2961	21	20	20	1	1	1	20
Cell Data	3656	21	20	20	13	8	5	20
Fuel	51058	21	20	20	17	3	9	20
Spambase	8549	21	20	20	91	5	4	20

3.2 Explainability

To quantify the explainability of EOP models, we have chosen four metrics based on selection criteria and recommendations provided in comprehensive surveys by Geng and Hamilton [12] and Lenca et al. [18] and designed to scoring rules $A \to B$. Bayes Factor (BF) and Lift (L) are simple metrics of high intelligibility, that have been shown to perform well at identifying relevant rules [18]. Additionally, we consider Normalized Mutual Information (NMI) for the properties described in [12] and because of its applicability to hierarchical models. Further, we use J-Score (J), a well-studied symmetric measure of interestingness that considers impact of positive and negative examples in data. The exact formulas used to compute those metrics are shown below.

$$BF(A \to B) = \frac{p(A|B)}{p(A|\overline{B})} = \frac{n_{AB} n_{\overline{B}}}{n_B n_{A\overline{B}}}$$

$$L(A \to B) = \frac{p(B|A)}{p(B)} = \frac{n \cdot n_{AB}}{n_A n_B}$$

$$J(A \to B) = p(A)\Big(p(B|A)\log\frac{p(B|A)}{p(B)} + (1-p(B|A))\log\frac{1-p(B|A)}{1-p(B)}\Big)$$

$$A = \wedge_{i=1}^{d} a_i$$

$$NMI(A \to B) = \frac{\Big(\sum_{i=1}^{d} p(a_i, b)\log_2\frac{p(a_i,B)}{p(a_i)p(b)}\Big)}{-\sum_{i=1}^{d} p(a_i)\log_2 p(a_i)}$$

These metrics have been originally designed for scoring single rules, but they can easily be adapted to handle hierarchical models like EOP. In the formula below, $\mathcal{M}$ denotes the model, D is the depth of the hierarchy, R_i represents the set of regions that are handled by classifier h_i. R_i has cardinality q_i. $R_i(x)$ denotes the event that a point x belongs to a region in R_i, while $C(h_i, x)$ denotes the event that h_i correctly classifies point x.

$$\begin{aligned}
\mathcal{M} &= \{(R_i = \{r_1 \dots r_{q_i}\}, h_i) \mid i = \overline{1, D}\} \\
&= \{\cup(A_i \to B_i) \mid i = \overline{1, D}\} \\
A_i &= \Big(\wedge_{j=1}^{i-1} \neg R_j(x)\Big) \wedge R_i(x) \qquad B_i = C(h_i, x)
\end{aligned}$$

The metric M for the model is computed as a linear combination of component Ms obtained for individual levels of the hierarchy that are visited during prediction, weighted by their corresponding support:

$$M(\mathcal{M}) = \sum_{i=1}^{D} p(A_i) M(A_i \to B_i)$$

Tables 6 and 7 summarize explainability scores of EOP and CART models obtained using previously described synthetic and real-world data sets. For synthetic data, we computed means, standard deviations and p-values from paired T-test, to determine whether the observed differences in performance are significant. Bayes Factor becomes numerically unstable whenever one of the components of the hierarchical model is fully homogeneous with respect to the output class distribution. It is reflected in the result tables with symbol "Inf", and we ignore the corresponding datasets in computing summary scores and in comparisons. The empirical results show that EOP indentifies more explainable regions of feature space with regularity, according to all metrics but J-Score. The difference in J-Scores observed on synthetic data does not appear statistically significant, and J-Score results for real-world data are mixed.

Explainability is useful in many practical applications. It is often the case in scientific research when understanding of the results is as important as discovering patterns. The goal of one such application is to determine whether a stem cell has been subjected to a treatment. The hope is that it could be determined using a set of measurements taken under a microscope, such as the area and perimeter of the cell, the stage of the cell cycle at the time of the observation, the generation the cell belongs to, as well as some other measurements.

Table 6: Metrics for CART and Nonparametric EOP - with decision stumps - on artificial data

	CART				EOP			
	BF	L	J	NMI	BF	L	J	NMI
DS1	Inf	0.005	0.223	0.018	3.109	0.016	0.262	0.440
DS2	Inf	0.010	0.236	0.052	1.818	0.048	0.136	1.093
DS3	1.160	0.014	0.267	0.014	1.372	0.019	0.009	0.337
DS4	1.620	0.004	0.005	0.048	1.498	0.038	0.400	0.559
DS5	1.454	0.008	0.113	0.062	2.826	0.027	0.146	0.703
DS6	1.445	0.007	0.148	0.041	1.719	0.013	0.096	0.785
DS7	4.181	0.008	0.195	0.033	4.875	0.027	0.265	0.854
DS8	Inf	0.010	0.236	0.052	1.818	0.048	0.136	1.093
DS9	Inf	0.008	0.198	0.051	2.143	0.024	0.670	0.369
Mean	*1.972*	*0.008*	*0.180*	*0.041*	*2.458*	*0.029*	*0.235*	*0.693*
Stdev	*1.340*	*0.003*	*0.081*	*0.016*	*1.398*	*0.013*	*0.199*	*0.289*
tTest	*0.012*	*0.001*	*0.252*	*0.000*				

Figure 7 shows the EOP model obtained after training on 5,000 data points evaluated on an equally large test set. In this case, the hierarchical model only identifies intervals (one-dimensional rectangles) of feature space in which data can be confidently discriminated, rather than multidimensional combinations. This behavior can be tuned using the EOP dimensionality regularization parameter λ.

The interpretation is that the cells with specific features falling within learned intervals can be safely classified as having been subjected to treatment. If, going from the top level of the EOP hierarchy, classification by cell area is inconclusive, cell generation and cycle time are considered. The intuition is that for small or very large cells it may be more difficult to determine whether treatment was applied or not, however, falling back on generation and then cycle time helps to provide a confident answer in many such cases. As a side note, the overall classification accuracy of EOP on this data is 77%, comparing favorably to 72% obtained with a random forest model.

Table 7: Metrics for CART and Nonparametric EOP on real data

	CART				EOP			
	BF	L	J	NMI	BF	L	J	NMI
MB	1.982	0.004	0.389	0.040	1.889	0.007	0.201	0.502
BCW	1.057	0.007	0.004	0.011	2.204	0.069	0.150	0.635
BT	0.000	0.009	0.210	0.000	Inf	0.021	0.088	0.643
V	Inf	0.020	0.210	-0.010	2.166	0.040	0.177	0.383
Mean	*1.520*	*0.010*	*0.203*	*0.010*	*2.047*	*0.034*	*0.154*	*0.541*

Another example involves a spam detection problem - we use the Spambase dataset from the UCI repository [9]. The data contains about 4,000 records and 57 features. EOP obtains a spam prediction accuracy of 80%, with the top three projections and the associated high confidence regions shown in 8. Each two-dimensional EOP region is depicted with a distinct color. The scatter plots show testing data resolved at subsequent levels of the EOP hierarchy.

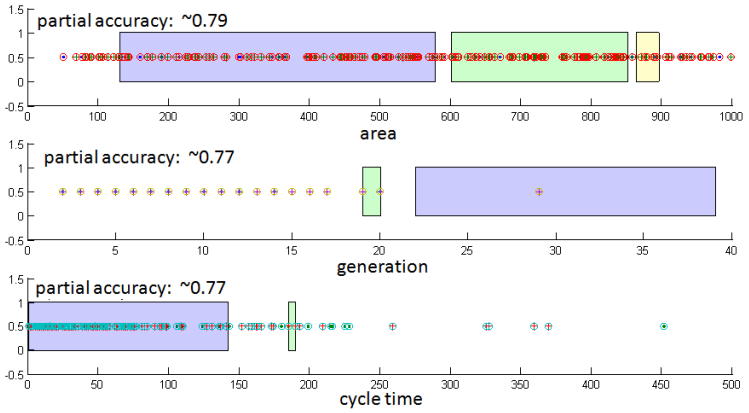


Figure 7: Explanatory projections for the Cell dataset

The classifier used in the first iteration simply labels everything as spam. The high confidence region, which indeed does enclose mostly spam test examples, does not have a high incidence of the word 'your', but it shows a high incidence of capital letters, which makes an intuitive sense. When the next iteration classifier is less likely to mark something as spam, the selected regions immediately reflect this semantic change: the threshold for the incidence of the word 'your' is lowered and the required incidence of capitals is increased. The square region on the left also encloses examples that will be marked as 'not spam' because of the lower incidence of capitals.

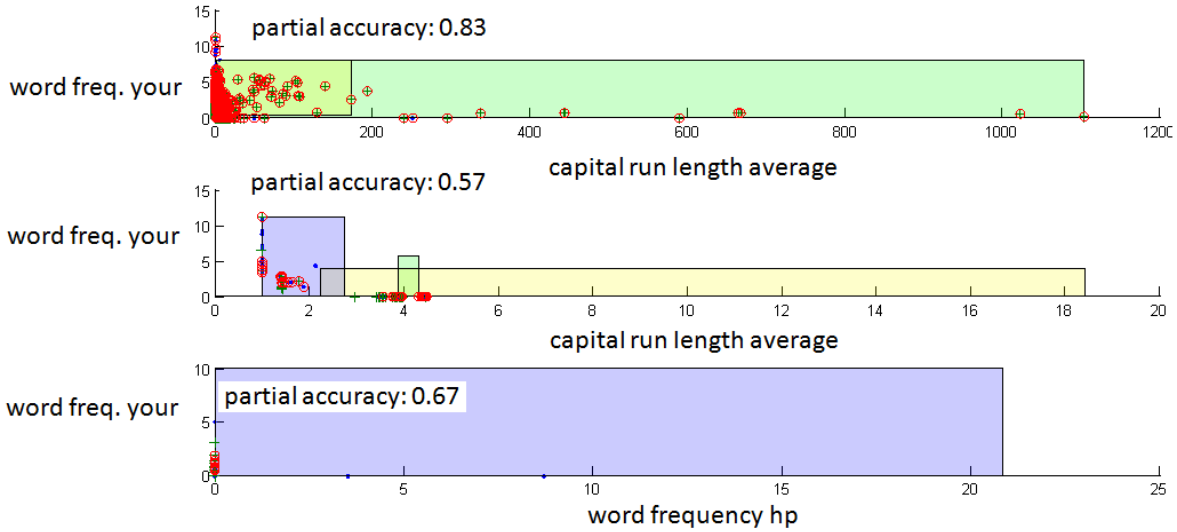


Figure 8: Explanatory projection for Spambase

3.3 Robustness

It is often useful in practice to identify subsets of data that are hard to confidently classify and set them aside. A variant of the EOP (Accuracy Targeting EOP, ATEOP), screens all projections of data for the largest robust regions where data can be classified with acceptable accuracy. Unlike the standard EOP implementation, ATEOP does not dynamically lower the error threshold to handle all data if possible. Instead, it aims at maintaining overall reliability of classification, and ignoring left-overs that are hard to deal with.

Algorithm 2 details ATEOP pseudocode. The parameter ϵ represents the allowable classification error rate, and α represents the minimum support of data that regions must provide in order to be considered worthy of inclusion in the model. Regions that meet these criteria on training data are verified using a separate validation set. If validation turns out too restrictive, ϵ is gradually reduced to force more robust selections during training. If multiple regions meet the threshold criteria, the one with the most extensive data coverage is chosen. The data in that region is then removed, and the process continues until all data is processed or no new satisfactory regions can be found.

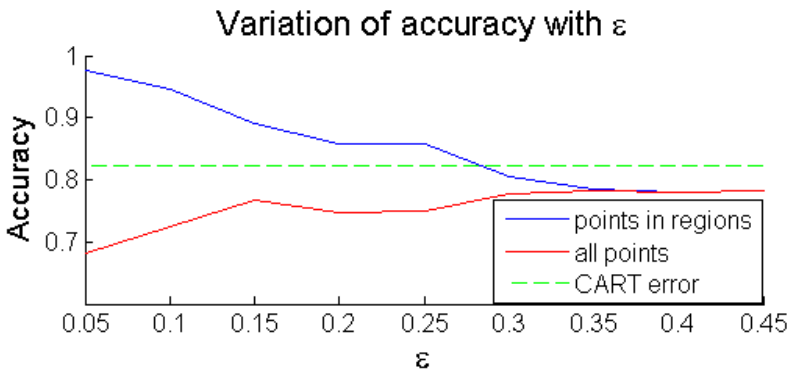


Figure 9: Accuracy of ATEOP a function of the allowable classification error rate: Accuracy for data covered by the model (blue); Accuracy for all data (red); Accuracy of the CART model on all data(green dashed)

Figure 9 displays the trade-off between achieving the required accuracy and data coverage. We ran ATEOP on one of the synthetic data sets explained above. In the graph we plot the obtained accuracy measured on the data included in the model (which is as high as required), as a function of the accuracy

threshold ϵ. We also plot accuracy of the default classifier applied to the left-over data. As ϵ increases, ATEOP is allowed to become more error-tolerant, the less data has to be left out, and the accuracy based on data included in the model goes down. Eventually, when accuracy threshold is lenient enough to allow all data to be included in some part of the ATEOP hierarchy, the two plots converge. For reference, also plot the accuracy achieved by CART. It is as expected slightly higher than the accuracy achieved at the convergence of the two ATEOP characteristics. A desired balance, which varies by application requirements, can be obtained through cross-validated selection of the threshold.

Algorithm 2 ATEOP Algorithm

```
ATEOP(ε,α)
ε_0 = ε
Classifiers=[]
Regions=[]
while TrainData is not Empty and foundProjection do
  foundProjection = false
  while ε_0 > 0 and not foundProjection do
    minRegionSize = α*size(TrainData)
    for all π ∈ Π do
      [h,R] = ObtainClassifierAndRegions(TrainData,ε_0)
      PointsInSet = R.filterPoints(CalibrationData)
      CalibrationError = h.classificationError(PointsInSet)
      if CalibrationError< ε
          and PointsInSet.size()>minRegionSize then
        minRegionSize = PointsInSet.size()
        Classifiers.add(h)
        Regions.add(R)
        foundClassifier=true
      end if
    end for
    if foundProjection then
      TrainingData.eliminatePointsIn(R)
      CalibrationData.eliminatePointsIn(R)
    end if
  end while
end while
```

3.3.1 Pattern Identification

A separate experiment illustrates the ability of EOP to identify patterns in data. Additional relatively small clusters of synthetic one class data from Gaussian distributions were injected into randomly chosen dimensions of the data.

The first column in the table in figure 10 shows which sets of features were impacted by the injections. This experiment involved 7 simultaneous injections containing about the same number of points. The columns of the table correspond to levels of the EOP hierarchy. Each cell i, j of the table shows how many of the injected data points belonging to

	Number of points picked at each iteration			
Pattern Features	Iteration 1	Iteration 2	Iteration 3	Default
[1,10]	31	5	3	1
[2,7]	36	5	4	0
[5,6]	44	8	1	0
[7,6]	34	10	5	0
[9,2,1]	29	10	1	0
[6,9,4]	21	12	4	0
[1,10,3,5]	41	7	3	0

Figure 10: Illustration of how EOP deals with injected lower-dimensional patterns - number of points from each pattern explained at each stage.

pattern i have been captured by some region at iteration j. The darker the color, the more points have been explained. Results show that EOP selects relevant projections of data at early iterations, to quickly reveal the injected overdensities - it deals with many of the datapoints at the very first iteration. Subsequently, the second projection explains another batch of points - corresponding to patterns 4,5 and 6. The first row of the table corresponds to a pattern spanning features 1 and 10. It consists of 40 data points. 31 of them were handled at level 1, the following 5 at level 2, and 3 at level 3 and 1 was left for the default classifier.

4 Conclusions

We have introduced Explanation-Oriented Partitioning, a data mining algorithm that learns explainable classifications. It works by identifying high confidence regions in low-dimensional projections of feature space that are populated by easy to classify data.

These regions can be used as contextual explanations to accompany predictions made for test queries. EOP can incorporate any externally provided classifiers. It relies on these to identify interesting projections of data that form a hierarchical, low-complexity model, that maintains competitive predictive accuracy while providing superior explainability of data when compared to relevant peers. The most important outcome however is that EOP classification results are easy to understand by human users. We have shown parametric and nonparametric variants of the procedure for identification of explainable regions of feature space.

The presented algorithm is shown to closely match the performance of boosting while providing completely explainable models. It also fares well when compared to alternative approaches by producing more compact models at a small tradeoff in accuracy. EOP algorithms are capable of finding expressive projections of data while maintaining high levels of fidelity. The resulting models are compact and capture the essence of data in the way that feels intuitive to users.

References

[1] L. Breiman. Stacked regressions. *Machine Learning*, 24:49–64, 1996. 10.1007/BF00117832.

[2] L. Breiman. Statistical modeling: The two cultures. *Statistical Science*, 2001.

[3] L. Breiman and L. Breiman. Bagging predictors. In *Machine Learning*, pages 123–140, 1996.

[4] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and Regression Trees*. Chapman and Hall/CRC, 1 edition, Jan. 1984.

[5] C. Cortes and V. Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.

[6] M. W. Craven and J. W. Shavlik. Extracting Tree-Structured Representations of Trained Networks. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 24–30. The MIT Press, 1996.

[7] P. Domingos. Knowledge discovery via multiple models. *Intelligent Data Analysis*, 2:187–202, 1998.

[8] E. M. Dos Santos, R. Sabourin, and P. Maupin. A dynamic overproduce-and-choose strategy for the selection of classifier ensembles. *Pattern Recogn.*, 41:2993–3009, October 2008.

[9] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[10] Y. Freund. Boosting a weak learning algorithm by majority, 1995.

[11] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting, 1997.

[12] L. Geng and H. J. Hamilton. Interestingness measures for data mining: A survey. *ACM Comput. Surv.*, 38, September 2006.

[13] T. K. Ho. The random subspace method for constructing decision forests. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(8):832 –844, aug 1998.

[14] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. In *Machine Learning*, pages 285–318, 1988.

[15] B. Liu, M. Hu, and W. Hsu. Intuitive representation of decision trees using general rules and exceptions. In *Proceedings of Seventeeth National Conference on Artificial Intellgience (AAAI-2000), July 30 - Aug 3, 2000*, pages 615–620, 2000.

[16] M. Mampaey, N. Tatti, and J. Vreeken. Tell me what i need to know: succinctly summarizing data with itemsets. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 573–581, New York, NY, USA, 2011. ACM.

[17] M. J. Pazzani, S. Mani, and W. R. Shankle. Beyond concise and colorful: Learning intelligible rules, 1997.

[18] P. M. B. Phillipe Lenca, B. V. A, and S. L. C. On selecting interestingness measures for association rules: user oriented description and multiple criteria decision aid, 2008.

[19] K. Sim, A. K. Poernomo, and V. Gopalkrishnan. Mining actionable subspace clusters in sequential data. In *SDM*, pages 442–453, 2010.

[20] P. Sollich and A. Krogh. Learning with ensembles: How over-fitting can be useful, 1996.

[21] L. B. Statistics and L. Breiman. Random forests. In *Machine Learning*, pages 5–32, 2001.

[22] K. Ting, J. Wells, S. Tan, S. Teng, and G. Webb. Feature-subspace aggregating: ensembles for stable andunstable learners. *Machine Learning*, 82:375–397, 2011. 10.1007/s10994-010-5224-5.

[23] G. Webb and Z. Zheng. Multistrategy ensemble learning: reducing error by combining ensemble learning techniques. *Knowledge and Data Engineering, IEEE Transactions on*, 16(8):980 – 991, aug. 2004.

[24] L. Wilkinson, A. Anand, and D. N. Tuan. CHIRP: a new classifier based on composite hypercubes on iterated random projections. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 6–14, New York, NY, USA, 2011. ACM.

[25] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.