Strategy Mining

Xiaoxi Xu, David Jensen and Edwina L Rissland
University of Massachusetts
Amherst, MA

Abstract

Strategy mining is a new area of research about discovering strategies for decision-making. It is motivated by how similarity is assessed in retrospect in law. In the legal domain, when both case facts and court decisions are present, it is often useful to assess similarity by accounting for both case facts and case outcomes. In this paper, we formulate the strategy mining problem as a clustering problem with the goal of finding clusters that represent disparate conditional dependency of decision labels on other features. Existing clustering algorithms are inappropriate to cluster dependency because they either assume feature independence, such as K-means, or only consider the co-occurrence of features without explicitly modeling the special dependency of the decision label on other features, such as Latent Dirichlet Allocation (LDA). We propose an Expectation Maximization (EM) style unsupervised learning algorithm for dependency clustering. Like EM, our algorithm is grounded in statistical learning theory. It minimizes the empirical risk of decision tree learning. Unlike other clustering algorithms, our algorithm is irrelevant-feature resistant, and its learned clusters modeled by decision trees are strongly interpretable and predictive. We systematically evaluate both the convergence property and solution quality of our algorithm using a common law dataset comprised of actual cases. Experimental results show that our algorithm significantly outperforms K-means and LDA on clustering dependency.

Introduction

Strategies play a key role in decision-making. In the context of decision-making, strategies involve how to evaluate decision-influential features and which decisions to make. Different people might use disparate strategies to make decisions. For example, physicians with different 'schools of medical thought' may prescribe different treatments for patients based on evaluating their previous medical complications, reported symptoms, and results of various tests, as shown in Figure 1. Courts may follow different doctrines in making their rulings, for instance, in our example, the relationship between buyer and seller of a product, or the harm caused by a defective product. Decision-making is also at the crux of political activity. In politics, different presidential administrations use different strategies to make decisions about war and peace, about budgetary funding priorities, and about which political candidate to support along with innumerable other choices. Organizations make far reaching strategic decisions about investment and direction of future growth; individual persons make decisions about how to live a life. When the outcome of a decision is observable (e.g., radical mastectomy leads to shorter or longer recovery time from breast cancer than lumpectomy does), uncovering the decision-making process is highly desirable. This is because strategies, once discovered, they can shed light on how to achieve a desired outcome and avoid an unwanted one, and can also enable strategy comparisons. This new area of research about discovering strategies in decision-making is what we call *strategy mining*.

In this paper, we formulate the strategy mining problem as a clustering problem, called the latent-strategy problem. In a latent-strategy problem, a corpus of data instances is given, each of which is represented by a set of features and a decision label. The inherent dependency of the decision label on the features, as shown in the above examples, is governed by a latent strategy. The goal is to discover the conditional dependency in order to reveal the strategy.

The difference between dependency-based clustering and conventional object-based clustering (Bishop 2006) is noteworthy. Object-based clustering deals with the joint distribution of all features in the feature space, whereas dependency-based clustering deals with the conditional distribution of the decision label on the other features. Existing clustering algorithms are inappropriate to cluster dependency because they either assume feature independence, such as K-means (MacQueen 1967) and mixture models, or only consider the co-occurrence of features without explicitly modeling the special dependency of the class label (i.e., decision label) on other features, such as LDA (Steyvers and Griffiths 2007)). In this paper, we propose a baseline algorithm for dependency clustering on the basis of the following assumption.

- *Data instances with similar features but different outcomes come from different conditional distributions.*

Our algorithm models conditional dependency with decision trees (Quinlan 1993) and iterates between an assignment step and a minimization step to learn a mixture of decision tree models that represent latent strategies. We call this algorithm Assignment Minimization for Decision Tree

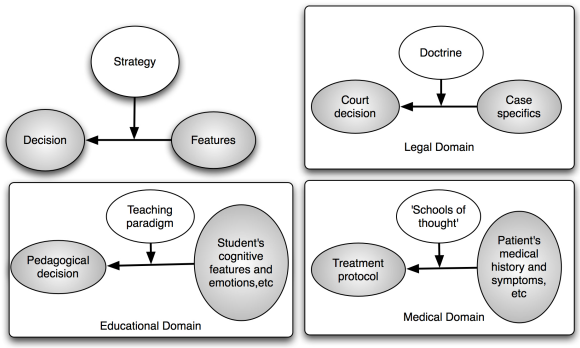


Figure 1: The relationship among strategy, decision labels, and features & examples in various domains

Mixtures (AM-DTM). Like the Expectation Maximization (EM) algorithm (Little and Rubin 2002) (Dempster, Laird, and Rubin 1977), AM-DTM is grounded in the PAC learning theory (Vapnik 2000). The difference, however, is that AM-DTM uses a non-parametric model, whereas EM-like algorithms often use parametric models.

AM-DTM is irrelevant-feature resistant, and its learned clusters (modeled by decision trees) are strongly interpretable and predictive.

1. *Irrelevant-feature resistant.*
   Unlike most clustering algorithms, AM-DTM is not sensitive to irrelevant features. This is because its contained decision tree models can intelligently select key features and identify how their interactions influence a decision.

2. *Strongly interpretable.*
   AM-DTM is a glass-box learning algorithm. The learned clusters are represented by decision trees, and therefore it would be easy to evaluate and explain the clustering results by examining the "look" of the trees.

3. *Predictive.*
   The learned clusters can be used as similarity measures to directly predict future data instances.

In this exploratory study, we evaluated the performance of AM-DTM using a real-world common law dataset. In this domain, there are 151 actual cases taken from a variety of jurisdictions in the United States and the United Kingdom. Our task is to learn the legal doctrines ruling the cases. The experimental results show that (1) AM-DTM converges quickly, (2) the learned decision tree models resemble disparate legal doctrines well, and (3) AM-DTM outperforms K-means and LDA on the task of clustering dependency.

The rest of the paper is organized as follows. In Section 2, we first show an example that motivates this paper and then define the latent-strategy problem. In Section 3, we present our baseline algorithm with a proof of convergence. Section 4 is devoted to the systematic evaluations of our algorithm. Related work is detailed in Section 5 and we conclude in Section 6.

The Latent-strategy Problem

A Motivating Example

Starting in 1852 and continuing for the next 60-80 years, a great change occurred in the doctrine governing the recovery by a remote buyer of a product that caused injury. Until the changes only a buyer who dealt directly with the manufacturer of the product – that is, who was said to be in 'privity of contract' – could recover for injury caused by the product. The old privity rule stated that if the buyer and maker of the product were not in a direct (privity) relation, there could be no recovery. The change in the doctrine was set into motion in 1852 by the landmark case of *Thomas and Wife v. Winchester*, (6 N.Y. 397 (1852)). In this case, under the existing privity doctrine, Mr. and Mrs. Thomas would not have been able to recover from Winchester because they did not buy the product directly from him (Levi 1949). However, the court decided to make an exception to the privity requirements in this case because the substance (bottle of poison (belladonna) mislabeled as 'dandelion extract') was so dangerous and the harm so severe. The court thus created an exception for things "imminently dangerous". This case set in motion the creation of a new doctrine that allowed for recovery from injuries caused by so-called "imminently dangerous" things even if the injured party and the maker of the goods were not in a direct contractual (privity) relationship. In 1916 another landmark case *MacPherson v. Buick Motor Co.,* (217 N.Y. 382, 111 N.E.1050 (1916)) was decided that completed the common law evolution of the doctrine. In this case, the wheel of an automobile fell off and injured the owner of the car (MacPherson). Since MacPherson had bought the car from a dealer, and not Buick directly, there was no privity. Also, a "car" is not in itself an 'imminently dangerous' article so it did not fit under the Thomas exception. At this point, The highest court in New York changed the law and rejected the requirement for 'privity' and the need to characterize a product "imminently dangerous" altogether, allowing for an injured party to recover for damage done by a defective product.

In the example above, there are three doctrines: (1) *privity for recovery*; (2) *imminently dangerous exception to the privity rule*; and (3) *recovery by remote buyers for defective products*. In Anglo-American law, it is often the case that the fading of an old doctrine requires the passage of time. The older doctrine of privity was not unfollowed by later cases immediately after the occurrence of the landmark case establishing the new doctrine. Rather, the privity doctrine faded as more cases were decided under the new doctrine. A similar situation occurred to the second and the third doctrine. We have also examined this domain for the role of exceptional cases in triggering doctrinal change (Rissland and Xu 2011). In this study, our goal is to uncover the different doctrines.

Problem Definition

Uncovering doctrines in the example above can be viewed as a clustering problem with the goal of finding clusters, each containing cases decided by the same doctrine. As a result, "similarity" in this domain should be interpreted as "similar strategies" that courts used to decide cases based on case

facts, or "similar conditional dependence" of court decisions on case facts.

This way of interpreting similarity is different from the conventional interpretation of similarity. Traditionally, similarity is often assessed by examining the joint distribution of all features in a nondiscriminating feature space. In this paper, we try to solve a clustering problem whose similarity is judged based on the class-conditional distribution in a discriminating feature space. We believe that some unobserved latent variables are responsible for how the *mappings* from features (e.g., case facts) to class labels (e.g., court decisions) are generated and, therefore, suggest that an EM-style process should be used to learn a generative model that specifies a joint probability distribution over the observed mappings and those latent labels.

After introducing the problem characteristics, we now define the latent-strategy problem. Examples are shown in Figure 1.

Definition 1. *In a latent-strategy problem, a corpus of data instances I is given, each of which is represented by a set of features F and a decision label D. The inherent dependency of the decision label on the features is governed by a latent strategy S. The objective is to find clusters, each containing data instances governed by the same strategy.*

A Baseline Algorithm on Clustering Conditional Dependency

- *Data instances with similar features but different outcomes come from different conditional distributions.*

As shown in Algorithm 1, AM-DTM, models conditional dependency with decision trees and iterates between an assignment step and a minimization step to learn a mixture of decision tree models that represent latent strategies.

Consider a dataset containing a set of data instances D. Each case has a set of features, among which there is a special class label feature. AM-DTM starts from partitioning D into K disjoint datasets $D_1 \dots D_k$, or K clusters. The partition can be done either with the guidance of domain knowledge or at random. Those K datasets are used to build K initial decision trees. Techniques should be used to avoid overfitting.

The main body of AM-DTM consists of two iterative steps: an assignment step (A-step) and a minimization step (M-step). In the A-step, instances are assigned to clusters based on decision tree classification results. The assignment strategy is as follows: if an instance in a cluster is correctly classified by the decision tree built from that cluster, it will stay in the original cluster; otherwise, it will move to a cluster whose decision tree correctly classifies it. When there is more than one decision tree that correctly classifies a misclassified instance, that instance will move to the cluster whose decision tree yields the highest classification probability for it. Further ties are broken by preferring the decision tree whose leaf node has a greater number of instances. If there is no decision tree that correctly classifies an instance, that instance stays in its original cluster.

In the M-step, decision tree learning is performed and the total training error of all decision trees is minimized under the assumption that the assignment from the A step is correct. To ensure and speed up convergence, we replace an older decision tree with a new one for a cluster only when the new tree has a lower training error. This process is repeated until no instance is moved. The goal of the iteration is to minimize the overall training error so that the learned decision trees representing coherent concepts can be found accurately.

Algorithm 1: AM-DTM

Input: D – Data instances; K – the number of clusters
Output: Learned decision trees DTs loaded with cases
Initialization: Randomly reshuffle cases in D; Partition D into K disjoint datasets $D_1 \dots D_K$
foreach D_i **do**
  Build a decision tree DT_i using proper strategies to avoid over-fitting
end
repeat
  A-step:
  foreach DT **do**
    $M \leftarrow$ find data instances that are misclassified
    foreach X *in* M **do**
      $DTree \leftarrow$ find the decision tree, over which X is correctly classified with the highest classification probability ;
      if $DTree \neq NULL$ **then**
        $D_j \leftarrow$ find dataset used to train $DTree$
        Move X to D_j
      end
    end
  end
  M-step:
  foreach D_i **do**
    Rebuild a decision tree $NewDT_i$ with proper strategies to avoid over-fitting
    $TrainError(NewDT) \leftarrow$ find the training error of the $NewDT_i$
    $MisclassificationRate(DT) \leftarrow$ find the misclassification error by evaluating DT_i on D_i
    if *TrainError(New* DT_i*)* $<$ *MisclassificationRate(*DT_i*)* **then**
      Replace DT_i with $NewDT_i$
    end
  end
until *convergence*;
return DTs

Theorem 1. *Using AM-DTM, the total training error strictly monotonically decreases until the algorithm converges.*

Proof. In each iteration of the AM-DTM algorithm, the misclassified data instances in one cluster are moved to another cluster only when they are correctly classified by the decision tree trained using that cluster. When the dataset of a cluster changes, a new tentative decision tree is created from the changed dataset. The decision tree for each cluster, how-

Table 1: Features and domain values in the common law dataset

Variables	Domain values
Outcome	P-win, D-win
Privity	Yes, No
Age	Infant/Youth, Adult
Plaintiff ID	Injured party, Representing injured party
Injury	Severe, Mild/ No injure
Injured-party role	Experienced user, Other
Defendant role	Manufacture, Vendor, Manufacture & Vendor, Other
Type	Food/drink, Drug/medicine, Chemicals, Personal care, Car, Machine, Other
IP role	Mere middleman or not
Existence of IP	Yes, No
Defect	Yes, No
Non-obvious defect	Yes, No
Inherently dangerous	Yes, No
Professional duty	Yes, No

ever, will be replaced by the new decision tree only when the new tree has a lower training error. Therefore, the total training error strictly monotonically decreases until the error reaches zero or no case are moved between clusters. □

Experiments and Results

Our evaluation dataset, as shown in the motivating example, consists of 151 actual cases taken from a variety of jurisdictions in the United States and the United Kingdom. Cases are represented by 14 features, which are somewhat different from those used in our previous work (Rissland and Xu 2011), see Table 1. We carried out 4 experiments to evaluate the performance of AM-DTM. In the first experiment, we evaluate the convergence property of AM-DTM; in the second experiment, we evaluate the AM-DTM's capability of discovering K; in the third experiment, we evaluate how well the learned decision trees resemble the three disparate legal doctrines; and in the last one, we compare the performance of AM-DTM with that of LDA and K-means.

Convergence

We ran AM-DTM 10 times with random initialization. As shown in the upper panel of Figure 2, the total error rate is plotted over the number of iterations. Error bars along the curve show the deviation across 10 runs. The monotonic decrease of the total error rate indicates the convergence of AM-DTM, which usually occurred within 5 iterations. The misclassification rate of each learned decision tree corresponding to each of the three clusters as a function of the number of iterations is shown on the lower panel of Figure 2.

Discovering K

In order to learn the number of latent legal doctrines, or the number of clusters K, we examined the sizes of learned decision trees and error rates in the training and cross-validation phases. The ideal K would be corresponding to K decision trees, each of which satisfies the following three requirements: (1) compact – because the doctrines are often

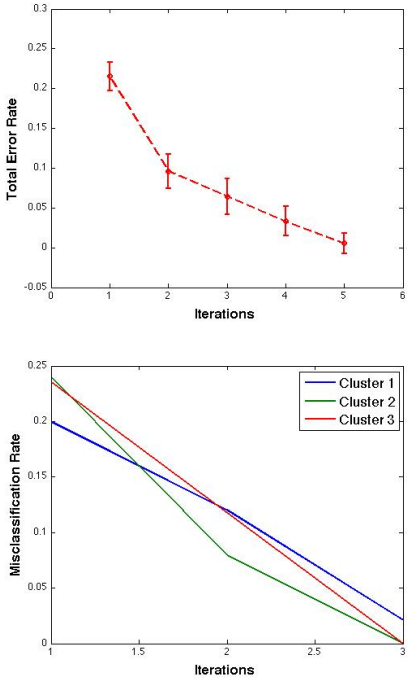


Figure 2: Total error rate across 10 runs (upper panel) & misclassification rate of different learned decision trees on 1 of 10 runs (lower panel)

not very complex legal rules; (2) low training error – because a cluster should represent a coherent strategy; and (3) low cross-validation error. When K=1, the learned decision tree has 20 nodes with a training error of 0.2 and a cross-validation error of 0.46. When K=2, the learned decision trees have an average of 7 nodes; the average training and cross-validation errors are 0.08 and 0.13, respectively. As shown in Figure 3, when K=3, the learned decision trees are compact with an average of 5 nodes and have the lowest average training error of 0.01 and the lowest average cross-validation error of 0.04.

Examining the learned decision trees

Recall that each case in the common law dataset was decided by one of the following three doctrines: (1) *privity for recovery*; (2) *imminently dangerous exception to the privity rule*; and (3) *recovery by remote buyers for defective products*. The fact that an older doctrine was not unfollowed by later cases immediately after the occurrence of a landmark case establishing the new doctrine presents a challenge to our clustering task because using landmark cases as cutoff lines to evaluate cluster membership would not work. Alternatively, we can evaluate the solution quality of AM-DTM by examining how well the decision trees learned by AM-DTM represent the three doctrines.

To show that the learned decision trees resemble the doctrines, we ran AM-DTM 10 times with random initialization and K set to 3. The top 5 features with the highest aver-

Table 2: The ranking list of the top 5 features with the highest average weights

(a) The Learned Decision Tree 1

Features	Weights
Product type	0.62
Defendant role	0.57
Privity	0.41
Defect	0.40
Injury	0.40

(b) The Learned Decision Tree 2

Features	Weights
Product type	0.79
Imminently dangerous	0.67
Defendant role	0.43
Non-obvious defect	0.37
Injury	0.36

(c) The Learned Decision Tree 3

Features	Weights
Product type	0.86
Defendant role	0.76
Imminently dangerous	0.39
Defect	0.38
Non-obvious defect	0.29

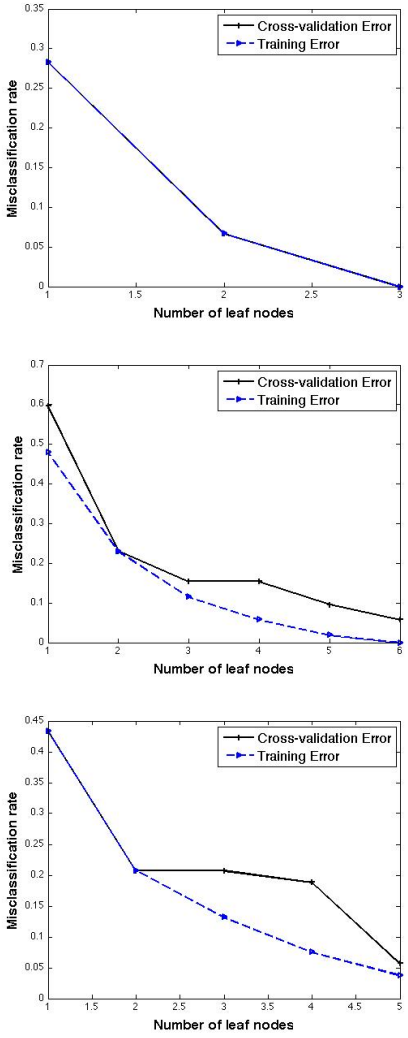
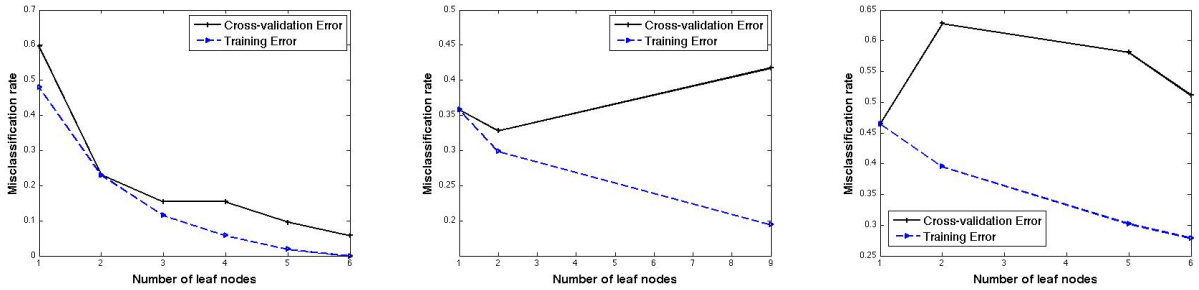


Figure 3: K=3, ten-fold cross-validation & training errors of decision trees built using each data cluster given by AM-DTM

age weights across 10 runs for each learned decision tree are shown in Table 2. A feature weight is defined as the ratio of the number of cases assigned to a feature node to the number of cases underneath the root node in the same tree. Overall, the ranking lists with feature weights resemble the doctrines well. For example, in the case of the second doctrine – 'recovery for imminently dangerous products', it is easy to explain what the doctrine is based on the corresponding ranking list. Specifically, 'imminently dangerous' (the second one on the ranking list) implies that only certain types of products (the first one on the ranking list, e.g., drugs) were considered to be imminently dangerous and therefore only certain defendant roles (the third on the ranking list) were considered to be liable for the injuries (the fifth on the ranking list) caused by defective products. The defects of an imminently dangerous product (e.g., drug) are often not obvious (the fourth on the ranking list), for example, a patient could hardly tell a dandelion extract (a harmless medicine) from an extract of belladonna (a deadly poison). The first and the third doctrines can be explained in a similar way based on their corresponding feature ranking lists. Note that the presence of "defendant role" in the first ranking list is due to the fact that defendant roles determine whether a case is a privity one or not (e.g., consumers when buying certain things from a grocery store are contracted with the vendor not the manufacture). The presence of "defendant role" in the third ranking list is because the *manufacturer* of a defective product was always considered to be liable for the injuries according to the third doctrine.

Comparing AM-DTM with K-means and LDA on clustering dependency

We also compared AM-DTM with K-means and LDA for clustering dependency. K-means and LDA are interesting comparisons because one (i.e., K-means) assumes feature independence and the other (i.e., LDA) only considers the co-occurrence of features without explicitly modeling the dependency of the decision label on other features. The basic idea of LDA is that documents are represented as random mixtures over latent topics, where each topic is characterized by a distribution over words. In our application, doctrines are mapped to topics in the text analysis, cases are mapped to documents, and case features are mapped to words. Moreover, the term frequency of a feature was set to 1 or 0 depending on the presence of that feature. As we can see from Figure 4, decision trees built from clusters discovered by K-means (K=3) and LDA (with alpha=17 and beta=0.01) have high bias and variance and are relatively complex. Specifically, both K-means and LDA have an average 0.2 training error and 0.5 cross-validation error. LDA produced decision trees that were slightly more compact (7 leaf nodes on average) than did K-means (10 leaf nodes on average). We were not surprised by the fact that LDA did not outperform K-means on clustering dependency because it did not model those desired dependency explicitly. Combining with the results from Figure 3, we can easily see that



(a) AM-DTM (b) K-means (c) LDA

Figure 4: Ten-fold cross-validation & training errors of decision trees built using 1 of the 3 data clusters given by (a) AM-DTM, (b) K-means, and (c) LDA. The selected clusters are representative to the general truth of all clusters.

AM-DTM significantly outperformed K-means and LDA in the task of clustering dependency.

Related Work

To the best of our knowledge, strategy mining is a new concept for data mining. No prior work has attempted to solve the latent-strategy problem we proposed in this paper. Because AM-DTM is a model-based clustering algorithm and uses an EM-style procedure to learn a non-parametric model, we focus on reviewing related work on the front of clustering analysis and the front of non-parametric EM algorithm for learning latent variables.

Comparing other works in clustering analysis, we think LDA is the one most similar to our work. Although LDA considers the co-occurrence of features, it does not explicitly model the special dependency of the decision label on other features. Most other clustering algorithms, such as K-means, often assume feature independence.

Although EM-style procedures are often used to learn parametric models, such as Gaussian mixtures, prior work has been done on using EM to learn non-parametric models, such as (Caruana 2001). That work presents an EM-style algorithm for learning a K-nearest neighbor (KNN) model to impute missing values in the data. However, the question of how to insure the convergence of non-parametric EM-style algorithm like KNN was left to open discussions. In this paper, we presented two mechanisms (i.e., when to move data and when to replace trees) to guarantee the convergence of AM-DTM for a mixture of decision tree models.

We think, most importantly, the main difference between those algorithms and AM-DTM is that they are not applicable to clustering conditional dependency.

Conclusion

Two objectives have been fulfilled by this paper. First, we defined latent-strategy discovery as a new problem for data mining. Its goal is to cluster conditional dependency. Second, we presented a baseline unsupervised algorithm to solve the problem. In our preliminary study of learning disparate legal doctrines from a real-world common law data set, experimental results show that our algorithm significantly outperforms K-means and LDA on the task of clustering dependency.

For future work, we will explore the use of other models to represent conditional dependency and compare their performance with the current algorithm using decision tree models. For example, we can choose other non-parametric statistical models (e.g., kernel Fisher discriminant analysis), parametric discriminative models (e.g., logistic regression), and parametric generative models (e.g.,naive Bayes).

References

Bishop, C. M. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. New York: Springer-Verlag.

Caruana, R. 2001. A non-parametric em-style algorithm for imputing missing values. *Artificial Intelligence and Statistics*.

Dempster, A. P.; Laird, N. M.; and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society* 1–38.

Levi, E. H. 1949. *An Introduction to Legal Reasoning*. Chicago: University of Chicago Press.

Little, R. J. A., and Rubin, D. B. 2002. *Statistical Analysis with Missing Data, Second Edition*. Wiley-Interscience.

MacQueen, J. 1967. Some methods for classification and analysis of multivariate observations. *Proc. of the Fifth Berkeley Symposium on Mathematics, Statistics and Probability* 1(5):281–296.

Quinlan, J. R. 1993. *C4.5: programs for machine learning*. San Francisco, CA: Morgan Kaufmann Publishers Inc.

Rissland, E. L., and Xu, X. 2011. Using case-based tests to detect gray cygnets. In *Case-Based Reasoning Research and Development*. Springer. 258–273.

Steyvers, M., and Griffiths, T. 2007. Probabilistic topic models. *Handbook of Latent Semantic Analysis* 22:424–440.

Vapnik, V. 2000. *The Nature of Statistical Learning Theory (Information Science and Statistics)*. Springer Verlag.