A Framework of Quantifying Subjective Unexpectedness of Pattern Measurements

Bo Kang

16th March 2015

Master's Thesis Computer Science

Advisor: Dr. Mario Boley

First Referee: Prof. Dr. Stefan Wrobel

Second Referee: Prof. Dr. Rainer Manthey

Institut für Informatik

Mathematisch-Naturwissenschaftliche Fakultät der

Rheinischen Friedrich-Wilhelms-Universität Bonn

Declaration

I hereby affirm that I have prepared the present paper self-dependently, and without the use of any other tools, than the ones indicated. All parts of the text, having been taken over verbatim or analogously from published or not published scripts, are indicated as such. The thesis has not yet been submitted in the same or similar form, or in extracts within the context of another examination.

Signed: _____

Date: _____

Abstract

By modeling user preferences, a data mining system is able to present mining results (patterns) that accommodate users' interests. However, as many state-of-art data mining systems lack any mechanism of measuring the amount of additional knowledge conveyed by mining results given what a user already knows, they might: a. provide patterns that are trivial according to the user's semantic knowledge, b. filter out patterns that are syntactically redundant but still convey knowledge that is new to the user. As a consequence, these data mining systems could be inefficient in delivering new knowledge according to different background knowledge of users. To address this issue, we introduce in thesis a knowledge model that approximates a user's current knowledge about a dataset and computes the so called *subjective unexpectedness* of pattern measurements (mining results). The computed subjective unexpectedness serves as an indicator that indicates the amount of additional knowledge conveyed by pattern measurements, in addition to what a user already knows. This thesis also describes the implementation of the knowledge model for a specific measurement type (i.e., frequency measurements) within a concrete data mining system, i.e., the One Click Mining System (OCM). To evaluate the effect of the knowledge model implemented in OCM, a user study is designed and conducted. The design and results of the study are documented in this thesis.

The knowledge model is potentially applicable to any data mining system in which the knowledge about datasets is conveyed to users via pattern measurements. In practice, the result of the user study confirms that the OCM system with the knowledge model provides users new knowledge more efficiently than the system without the knowledge model. We expect to generalize the knowledge model such that it is applicable to different data types. In order to efficiently compute in practice the subjective unexpectedness for more pattern measurement types, various implementation techniques remain to be investigated.

Acknowledgments

I would like to express my gratitude to my advisor, Dr. Mario Boley, for his patience and guidance throughout my master's study since 2012. I genuinely appreciate his expertise and his principles of work.

My thanks also goes to Prof. Dr. Stefan Wrobel and Dr. Tamás Horváth for introducing me into this exciting field of Machine Learning and Data Mining.

This thesis was written in the research group CAML at the Knowledge Discovery Department of Fraunhofer IAIS in Sankt Augustin. I would like to thank Dr. Thomas Gärtner, the head of CAML group, who offered me this great opportunity of working as a research assistant in the group.

I thank my colleague Björn Jacobs, who has greatly helped me in my thesis' implementation.

Last but not the least, I would like to thank Dr. Tijl De Bie for his patience of answering my questions regarding to his previous work of subjective unexpectedness.

This work was supported by the German Science Foundation (DFG) under the reference number 'GA 1615/2-1'.

Wir müssen wissen, Wir werden wissen!

David Hilbert

Contents

1 Introduction **3**
- 1.1 Background and Motivation . . . 3
- 1.2 Contributions . . . 4
- 1.3 Outline . . . 4

2 Preliminary **5**
- 2.1 Notations . . . 5
- 2.2 Pattern Ranking Mechanism in OCM . . . 6
- 2.3 Subjective Unexpectedness for Patterns . . . 7

3 Subjective Unexpectedness of Pattern Measurements **9**
- 3.1 Constraints Induced by Measurements . . . 9
- 3.2 Maximum Entropy Model . . . 10
- 3.3 Computing Subjective Unexpectedness of Pattern Measurements . . . 12
- 3.4 Computing Subjective Unexpectedness of Frequency Measurements . . . 13

4 An Efficient Implementation of Knowledge Model **16**
- 4.1 Computing the Partition Function $Z(\boldsymbol{\lambda})$ . . . 16
- 4.2 Applying the Knowlege Model in OCM . . . 17
  - 4.2.1 Computing Subjective Unexpectedness . . . 18
  - 4.2.2 Using Subjective Unexpectedness . . . 18

5 User Study **20**
- 5.1 Study Design . . . 20
  - 5.1.1 Task Specification . . . 21
  - 5.1.2 System Performance Metric . . . 23
  - 5.1.3 Assignment Logic . . . 24
- 5.2 Study Implementation . . . 25
- 5.3 Study Conclusion . . . 25

6 Conclusion **26**
- 6.1 Discussion . . . 26
- 6.2 Outlook . . . 26

A Maximum Entropy Model **29**
- A.1 The MaxEnt Problem is a Concave Optimization Problem . . . 29
- A.2 Equivalence of the Lagrange Primal Problem . . . 30
- A.3 Concavity of the Lagrangian w.r.t Multipliers $\boldsymbol{\lambda}$, μ . . . 30
- A.4 Convexity of the Lagrangian w.r.t Distribution Function $P(\cdot)$ . . . 30
- A.5 Derivation of the Background Distribution . . . 31
- A.6 Derivation of the Dual Function . . . 32

A.7 Concavity of the Lagrange Dual Function Parametrizied Only by $\boldsymbol{\lambda}$ 33
A.8 An Example of Computing Unexpectedness of Frequency Measurements 34

B User Study **36**
B.1 Data Generating Process 36
B.2 Analysis Task Instructions 37
B.2.1 Analysis Task Description 37
B.2.2 System Instruction 38
B.3 Rating Task Instruction 39
B.4 Intermediate Sutdy Measurments 40

Chapter 1

Introduction

1.1 Background and Motivation

Many state of art data mining systems (e.g. WEKA [Hall et al., 2009], Rapid Miner [Mierswa, 2009]) require the user to construct a workflow with an explicit selection of data mining methods, parameters and algorithms. This adds too much overhead to the users who are not data mining experts. To eliminate the user's burden on constructing an efficient data mining workflow, Boley et al. [2013] proposed the so called "One Click Mining" (OCM) framework. The framework models user preferences of mining results (patterns) by interpreting user-systems interaction as users' implicit feedback. Based on the user preference model, OCM constructs a user-oriented mining process by automatically operating the mining algorithms and ranking the mined results. When exploring a dataset by interacting with OCM, a user gains knowledge about the dataset via patterns. Syntactically a pattern is represented by two parts: the description of a pattern that gives a statement about the underlying dataset and different types of pattern measurements that indicate numerically how interesting a pattern is according to some predefined measure functions. Hence modeling user preferences is essentially approximating a user's interests of the syntactical knowledge conveyed by patterns.

While OCM provides mined patterns with user preferred syntax, it lacks any mechanism of measuring the amount of additional knowledge conveyed by patterns in addition to what the user already knows, i.e., the semantic knowledge of patterns. This has two negative consequences: first, the OCM framework might present the patterns with high utilities but are trivial according to a user's current knowledge; second, it might filter out the patterns with certain syntactical redundancy but are potentially interesting to the user when constructing pattern rankings. This means the OCM system can be inefficient in delivering new knowledge according to different background knowledge of users. Hence, to address this issue, we need to capture a user's semantic knowledge about the dataset that the user is working on and measure the new knowledge conveyed by patterns based on the user's current knowledge state. De Bie [2009] proposed a model to quantify such new knowledge by measuring the so called *subjective unexpectedness* of patterns. The model represents the user's knowledge state by a so called *background distribution* which is retrieved from a constrained family of distributions by applying the max entropy principle. The unexpectedness of a pattern is then determined by trading off two factors: the probability of the pattern's appearance under the background distribution and the pattern's complexity. However, as in OCM knowledge is conveyed by both pattern descriptions and pattern measurements, further modeling a user's unexpectedness with respect to description and measurement is required.

Motivated by this requirement, this thesis investigates a general framework of measuring subjective unexpectedness of pattern measurements in any data mining system that conveys knowledge about datasets (partially) via pattern measurements, and refers to this framework as the *knowledge model*. We leave modeling unexpectedness for pattern descriptions as a future work. The thesis also explores possible efficient implementations of the knowledge model. Finally, this thesis presents a user study that evaluates the effects of applying the knowledge model in OCM.

1.2 Contributions

There are three contributions of this thesis:

1. *Investigating an framework of quantifying subjective unexpectedness for general pattern measurements (the knowledge model).* Instead of measuring subjective unexpectedness of a pattern, we turn to measure the unexpectedness for different types of pattern measurements. We quantify the subjective unexpectedness of a pattern measurement as the difference between the measurement observed in current working dataset and the user's expected measurement of the same pattern under the background distribution. This framework is potentially applicable to any data mining system in which the knowledge about datasets is partially conveyed to users via pattern measurements. This framework is discussed in Chapter 3.

2. *Developing an algorithm that efficiently computes the subjective unexpectedness with respect to a special type of pattern measure, i.e., the frequency measure.* In general, the complexity of computing unexpectedness of pattern measurement is exponential in terms of the input dataset size. However, the subjective unexpectednesses of frequency measurements can be computed with complexity that is exponential in the number of user investigated patterns. As the cognitive energy of a user is limited, for each analysis task, the number of patterns that the user is willing to investigate is usually small. Hence in practice our algorithm is scalable in terms of dataset size and computes the subjective interestingness of frequency measurements more efficiently than the general approach. This algorithm is described in Chapter 4.

3. *Designing a repeatable and scalable user study that evaluates the benefit of applying the knowledge model in OCM.* We applied the algorithm of computing subjective unexpectedness for frequency measurements in OCM (in Chapter 4). According to the motivation of the knowledge model, we defined a hypothesis: "By quantifying the subjective unexpectedness of pattern measurements, the OCM system provides a user additional knowledge (in additional to what the user knows) more efficiently than the OCM system that does not measure subjective unexpectedness." To evaluate this hypothesis, we developed a study design which serves as a protocol for testing our hypothesis. According to the study design, the hypothesis can be repeatedly evaluated by new study instances in the future with respect to any number of study participants. Based on the the study design, we conducted a study instance whose result justified our hypothesis. The study design as well as the conducted study instance are described in Chapter 5.

1.3 Outline

After a brief introduction of some basic notions, the pattern ranking mechanism of the OCM framework and the main idea of modeling the model the subjective unexpectedness for patterns in the existing works will be discussed in Chapter 2. The knowledge model is described in Chapter 3. In Chapter 4, we discuss the algorithm that computes subjective unexpectedness for frequency measurements. The implementation of the algorithm in OCM is also described in this chapter. In Chapter 5, we present how a user study is designed and conducted for evaluating the effect of the knowledge model in OCM. The future works are listed in Chapter 6.

Chapter 2

Preliminary

This chapter introduces the basic notions and concepts used in this thesis. Particularly, it introduces the pattern recommendation mechanism in the OCM system and motivates the requirement of measuring subjective unexpectedness. The main idea of modeling the subjective unexpectedness for patterns according to De Bie [2009] is also recapped in this chapter.

2.1 Notations

Denote data space $\mathbb{D}$ as a set of all possible datasets. A dataset $D \in \mathbb{D}$ consists a set of data records $r \in R$. Each data record is described by a fixed attribute set A, where each attribute $a \in A$ maps the record to a value domain $\mathbb{V}_a$, i.e., $a : R \to \mathbb{V}_a$. In this thesis we only focus on to categorical dataset, i.e., $|\mathbb{V}_a|$ is finite and its values are incomparable. Given an attribute set $A = \{a_1, \ldots, a_{|A|}\}$, each data record $r \in R$ corresponds to a value vector $\boldsymbol{v}_r = \big(a_1(r), \ldots, a_{|A|}(r)\big)$. The value space of value vectors is denoted as $\mathbb{V} = \times_{a \in A} \mathbb{V}_a$.

For each data record r in a dataset D, a proposition o is a binary statement which holds either true or false about the record, i.e., $o : \mathbb{V} \to \{0, 1\}$. Given a proposition set $\mathbb{O}$, a pattern language $\mathcal{L}$ consists of logic combinations of the propositions from the set. For instance, the OCM framework produces association patterns (e.g. fig.2.1a) and subgroup patterns (e.g. fig.2.1b), for the association patterns we have pattern language consists of conjunctions of the propositions, i.e., $\mathcal{L}_{\text{cnj}}$; and for the subgroup patterns, the pattern language consists of conjunctions of propositions combined with the index set of the target attributes, i.e., $\mathcal{L}_{sgd} = \mathcal{L}_{cnj} \times [n]$. For each element s in a pattern language $\mathcal{L}$ is referred to as a pattern descriptor. An association pattern descriptor $s_{ass} \in \mathcal{L}_{\text{cnj}}$ has form $s_{ass} = (o_1 \wedge \ldots \wedge o_k)$, while a subgroup descriptor $s_{sgd} \in \mathcal{L}_{sgd}$ is defined as $s_{sgd} = (o_1 \wedge \ldots \wedge o_k, t)$ where $t \in [n]$. Denote $\mathcal{F}$ as a set of predefined interestingness measure functions. That is, given a dataset, an interestingness measure $f \in \mathcal{F}$ maps a pattern described by s into a real value, i.e., $f : \mathcal{L} \times \Omega \to \mathbb{R}$. For example,

- The *frequency measure* for the association pattern is defined as:

$$f_{freq}(s_{x_{ass}}, D) = \frac{|support(s_{x_{ass}}, D)|}{|R_D|}, \tag{2.1.1}$$

  where $support(s_{x_{ass}}, D)$ is the set of rows in dataset D that their values satisfy the descriptor $s_{x_{ass}}$ of pattern x_{ass}.

- The *lift measure* [Boley et al., 2013] computes the difference between the frequency of an association pattern and its expected frequency according to the assumption that all propositions within the descriptor are mutually independent, i.e,

$$f_{\text{lft}}(s_{ass}, D) = \Big(f_{freq}(s_{ass}, D) - \prod_{o \in \mathbb{O}_s} f_{freq}(s_o, D)\Big)/2^{|\mathbb{O}_s|-2} \tag{2.1.2}$$

where $\mathbb{O}_s$ is the set of propositions that present in descriptor $s_{x_{ass}}$, and s_o denotes a descriptor that contains only one proposition o.

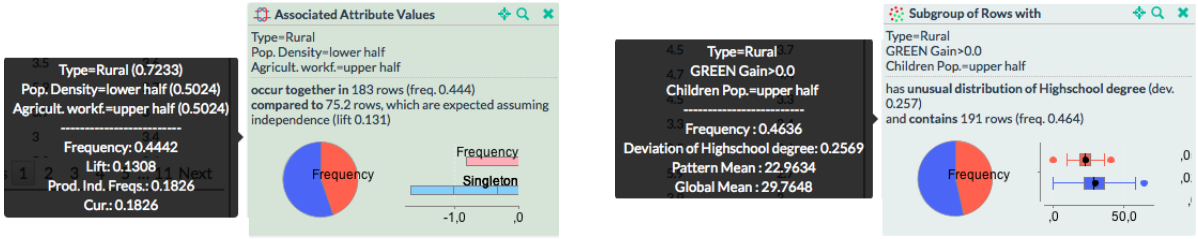
- The *subgroup measure* $f_{\text{sgd}}(s_{x_{\text{sgd}}}, D)$ ([Boley et al., 2013]) of a subgroup pattern x_{sgd} is defined as the multiplication of frequency $f(s_{x_{\text{sgd}}}, D)$ and *target deviation* measure $f_{\text{dev}}(s_{x_{\text{sgd}}}, D)$, i.e.,

$$f_{\text{sgd}}(s_{x_{\text{sgd}}}, D) = f_{\text{freq}}(s_{x_{\text{sgd}}}, D) f_{\text{dev}}(s_{x_{\text{sgd}}}, D) \tag{2.1.3}$$

  where $f_{\text{dev}}(s_{x_{\text{sgd}}}, D)$ is defined as the distance between the distribution of target attribute a_t in the dataset extension described by $s_{x_{\text{sgd}}}$ and in the whole dataset.

Corresponding to a measure f, a measurement m with respect to descriptor s and data D is defined as a pair $m = (f, \alpha)$, where $\alpha \in \mathbb{R}$, $f(s, D) = \alpha$.

A pattern p is a pair $(s, M) \in \mathcal{L} \times 2^{\mathcal{F} \times \mathbb{R}}$, where the measurement set M is associated with $F \subseteq \mathcal{F}$, a set of applicable interestingness measure according to the pattern type. For example, to evaluate the interestingness of the association patterns in OCM, frequency measure and lift measure are used, hence the association pattern in (Figure 2.1a) has a frequency measurement $f_{\text{freq}} = 0.4442$ and a lift measurement $f_{\text{lft}} = 0.1308$. For subgroup patterns, besides the subgroup measure, the frequency measure is also applied. As a concrete example, the subgroup pattern in (Figure 2.1b) has a subgroup interestingness measurement $f_{\text{sgd}} = 0.2569$ and also a frequency measurement $f_{\text{freq}} = 0.4636$. For different pattern types, the measure set F varies. A pattern $p = (s, M)$ is said to occur in the data D if for every $(f, \alpha) \in M$ we have $f(s, D) = \alpha$.



(a) Association pattern with measures: *frequency* and *lift*

(b) Subgroup pattern with measure: *frequency* and *target deviation*

Figure 2.1: Patterns that are discovered via OCM

2.2 Pattern Ranking Mechanism in OCM

In order to recommend the patterns that might be interesting to a user, patterns discovered in OCM are ranked according to the modeled user preference. This is achieved in three steps. First, a pattern x is characterized by a predefined set of features Φ. There are two types of features in OCM, a real valued feature $\varphi_f(x) \in \mathbb{R}$ is induced by a pattern measure, i.e., $\varphi_f(x) = f(x)$; a binary feature $\varphi_o(x) \in \mathbb{B}$ indicates whether a particular proposition o is contained in the descriptor of pattern x. Denote $\mathbb{F}$ as the linear feature space induced by the feature set Φ, each pattern has its corresponding feature vector $\boldsymbol{\varphi}(x) = (\varphi_1(x), \ldots \varphi_{|\Phi|}(x))$, where $\boldsymbol{\varphi}(x) \in \mathbb{F}$. Second, defined on the feature representation of the patterns, a user preference model $u : \mathbb{F} \to \mathbb{R}$ is maintained in the OCM framework. For each pattern $x = (s, M)$ that is produced in OCM, the pattern utility $u(\boldsymbol{\varphi}(x))$ serves as an indicator of the user preference of this pattern. Third, given the feature space and the user preference model, the ranking of a newly discovered patterns set Q can be increasingly constructed: greedily add into the ranking with the pattern which has the largest minimum weighted cosine distance to the patterns within the ranking in the feature space, formally:

$$r' = r \cup x, \text{where } x = \underset{x \in Q}{\text{argmax}} \min_{p' \in r} u(\boldsymbol{\varphi}(x)) \cdot \text{dist}_{\cos}(x, p') \tag{2.2.1}$$

Since both the cosine distance and the pattern utility are bonded with the feature representations of the patterns, the current ranking mechanism selects the pattern with descriptors and measures that are the most syntactically dissimilar to the ranked patterns, it maintains the syntactical diversity of a ranking. However, because for different users, their semantic knowledge about even the same dataset vary, by maintaining merely the syntactic diversity in a ranking can not guarantee to present different users always new knowledge in addition to their background knowledge. We define *semantic diversity* as the quantity that indicates the amount of new knowledge in a ranking with respect to a user's current knowledge state. The following cases show that the semantic diversity is more powerful than the syntactic diversity, and can be used to distinguish more complex relationships among the patterns, while the later one fails to tell:

- For a new pattern that is syntactically redundant, it also conveys no new semantic knowledge. For example, the new pattern that is exactly as the same as one of the previously studied pattern.
- A pattern that is complementary to the user already learned patterns, is syntactically new but semantically redundant. For instance, in the German social-economical dataset which we compiled from the database provided by the German Federal Office of Statistic[1], an association pattern with descriptor **(area = rural ∧ tractor number = high)** is complementary to the pattern described by **(area = urban ∧ tractor number = low)**, hence these two patterns have different syntaxes. But these two patterns describe the same information in from two logic perspective, hence they are semantically redundant.
- New pattern with descriptor that is covered by as set of patterns known by user, but is not exactly the same as any previous one. In this case, the new pattern has certain syntactic diversity but the new semantic knowledge contained in the pattern varies: a. the new pattern can be derived from the previous pattern, hence is not interesting, e.g., the pattern with descriptor **(area = rural ∧ tractor number = high)** can be concluded by the prior knowledge **(tractor number = high ∧ agriculture workforce rate = high)** and **(area = rural ∧ agriculture workforce rate = high)**; b. the new pattern reveals additional information upon the studied patterns. For example, given patterns with descriptors **(area = rural ∧ high school degree holder rate = low)** and **(tractor number = high ∧ crime rate = low)** that have been studied by user, the next pattern with descriptor **(area = rural ∧ tractor number = high)** is syntactically covered by previous two patterns, but it tells the knowledge that can not be directly inferred from the previous ones.

The subjective unexpectedness measures the amount of the new information conveyed by the patterns according to the user, therefore it can be used as semantic diversity indicator when constructing the pattern ranking.

2.3 Subjective Unexpectedness for Patterns

To measure subjective unexpectedness of patterns that are discovered in a dataset, one need to first to model users' current knowledge about the dataset, i.e., *background knowledge*, denoted as $\mathcal{B}$. According to De Bie [2011], a user's knowledge about a dataset can be represented by a *background distribution* over the data space $P_{\mathcal{B}} : \mathbb{D} \to [0, 1]$. That is, the users belief state of how every dataset $D \in \mathbb{D}$ is likely to be the dataset that she is currently working on. As the knowledge about a dataset is essentially conveyed by pattern measurements, the background distribution $P_{\mathcal{B}}$ of a user is therefore characterized by the pattern measurements that has been observed by the user. That is, these observed measurements pose a set C of constraints on the possible background distributions over the data space [De Bie, 2009, Kontonasios et al., 2013].

However, as the constraint set does not uniquely determine the background knowledge in general, it results in a constrained family of distributions $\mathcal{P}_C$, formally:

$$\mathcal{P}_C = \{P : \mathbb{D} \to [0, 1] \mid \forall c \in C, c(P) = 1\} \tag{2.3.1}$$

[1] www.regionalstatistik.de

In order to represent the user's belief state as a unique background distribution, the *maximum entropy principle* (MaxEnt) is applied to retrieve the distribution with maximum entropy from the family $\mathcal{P}_C$ [De Bie, 2009]. By applying this principle, we obtain the background distribution associated with user's current background knowledge $\mathcal{B}$, i.e.

$$P_{\mathcal{B}} = \underset{P \in \mathcal{P}_C}{\operatorname{argmax}} - \mathbb{E}_{D \sim P} \log(P(D)) \tag{2.3.2}$$

The reason of applying the maximum entropy is that a distribution with lower entropy injects additional assumptions and hence is biased in undue ways. Therefore, the distribution with maximum entropy is the best choice at hand.

With the retrieved background distribution, the user's belief of the appearance of pattern x in dataset D is defined as the probability:

$$P_{\mathcal{B}}(x) = \sum_{D \in \mathbb{D}_x} P(D) \tag{2.3.3}$$

where $\mathbb{D}_x$ is the set of datasets that contain pattern x.

Finally, the subjective unexpectedness of a pattern x is defined by trading off the information content of the pattern (i.e., $-\log(P_{\mathcal{B}}(x))$) and the pattern's descriptional complexity, i.e., the *information ratio* of a pattern([De Bie, 2013]):

$$\text{InfoRatio}(x) = \frac{\text{InformationContent(x)}}{\text{DescriptionalComplexity(x)}}. \tag{2.3.4}$$

intuitively, with fixed descriptional complexity, if a pattern has a small appearance probability according to a user's belief state, then this pattern has a large information content, which indicates the information ratio of this pattern is also large. Hence the information ratio gives a reasonable indication of the subjective unexpectedness of a pattern.

In this thesis, in order to further quantify subjective unexpectedness for pattern measurements, we need to extend the model described above. This extended model is described in the next chapter.

Chapter 3

Subjective Unexpectedness of Pattern Measurements

This chapter describes a model that computes subjective unexpectedness for pattern measurements. First we give a detailed derivation of the approach proposed by De Bie [2013] that approximates user background knowledge and compute subjective unexpectedness for patterns. Then we extend this approach in order to compute subjective unexpectedness for pattern measurements. This extended approach is also referred to as the *knowledge model*, which is summarized as an algorithm in this chapter. Theoretically, the subjective unexpectedness of any types of pattern measurements (with form defined in 2.1) can be computed by the knowledge model. In practice, such computation can be very expensive. However, a better computational efficiency can be achieved for frequency measurements in particular. The knowledge model for frequency measurements is discussed in the end of this chapter.

3.1 Constraints Induced by Measurements

As previously discussed in section 2.3, to model a user's knowledge about the dataset that she is currently working on, one way is to represent user's knowledge as a background distribution $P_{\mathcal{B}}$ over data space $\mathbb{D}$. Such background distribution is characterized by a set of linear constraints converted from the observed measurements [De Bie, 2009]. Formally, for a pattern x in a given dataset D_{obs}, a pattern measurement $m_x = f(s_x, D_{\text{obs}})$ induces a linear constraint:

$$\mathbb{E}_{D\sim P,\mathbb{D}}[f(s_x, D)] = f(s_x, D_{\text{obs}})^1. \tag{3.1.1}$$

. Intuitively, once a user studied a measurement $f(s_x, D_{\text{obs}})$, the knowledge contained in the measurement is also conveyed to her, hence the future appearances of this measurement in dataset D_{obs} become expectable to the user. More importantly, this learning behavior changes the user beliefs about the datasets that might contain pattern x and have the same measurement $f(s_x, D_{\text{obs}})$. Thus, within those datasets, the user's beliefs about the appearances of other patterns (especially the ones that are correlated with x) will also change. Relating this intuition back to our model, constraints in form (3.1.1) asserts that a feasible background distribution should give in expectation the same values as the studied pattern measurements. Such distribution also assigns higher beliefs to the datasets that are more likely to give the observed measurements. As a consequence, the (non-revealed) measurements of those datasets also receive higher beliefs. Hence, the background distribution "learns" some knowledge conveyed by the constraints.

In general, a background distribution is characterized by a set of constraints. We denote the set of all possible distributions over data space $\mathbb{D}$ as $\mathcal{P}$, where each distribution $P \in \mathcal{P}$ is a mapping $P : \mathbb{D} \to \mathbb{R}$.

[1] Although data space $\mathbb{D}$ can be either discrete or continuous, in this thesis we only consider discrete data space. Hence computing the expectation of certain measure $f(\cdot)$ under distribution P is done by summing "$\sum$" over data space $\mathbb{D}$. When dealing with continuous data spaces, one need to replace the summation by integration "$\int$"

Given a pattern set X_{obs} that has been studied by a user, a set of constraints C in the form of (3.1.1) are induced by the corresponding pattern measurements M_{obs}. Then, the distribution set $\mathcal{P}_C \subseteq \mathcal{P}$ is referred to as the set of *feasible distributions* that satisfy all constraints in C. However, as a user only have one belief state at a moment, the subjective unexpectedness of a measurement also needs to be computed against a unique background distribution. That means, we need to further retrieve exactly one distribution out of the constrained set $\mathcal{P}_C$ according to certain object, i.e., distribution with the maximum entropy.

3.2 Maximum Entropy Model

To obtain a unique background distribution from a constrained distribution set $\mathcal{P}_C$, we follow the idea proposed by De Bie [2009] to search for the distribution function with the maximum entropy in $\mathcal{P}_C$. Then our problem can be transformed into an optimization problem whose objective function is entropy function $-\mathbb{E}_{D\sim P,\mathbb{D}}[\log P(D)]$ subject to the constraint set C. In this section we first formalize this optimization problem, and then discuss how to solve it by applying the *Lagrange method*. We show that our optimization problem fulfills the so-called *strong duality* property, which guarantees that the global optimum can always be found via the Lagrange method.

We formalize the optimization problem that maximizes the entropy (the MaxEnt problem) of a distribution function $P \in \mathcal{P}$ subject to a constraint set C as follow:

$$\max_{P\in\mathcal{P}} \quad g_0 = -\mathbb{E}_{D\sim P,\mathbb{D}}[\log P(D)] \tag{3.2.1}$$

$$\text{s.t.} \quad g_c(P(\cdot)) = \mathbb{E}_{D\sim P,\mathbb{D}}[f_c(s_{x_c}, D)] - f_c(s_{x_c}, D_{\text{obs}}) = 0, (\forall c \in C), \tag{3.2.2}$$

$$h(P(\cdot)) = \sum_{D\in\mathbb{D}} P(D) - 1 = 0. \tag{3.2.3}$$

The above problem is a *concave optimization problem* with a convex domain $\mathcal{P}$ and convex constraint functions (see Appendix A.1). It has a global optimal solution (denoted as $P^*_{\mathcal{B}}$) that maximize the objective function (3.2.1). According to [Boyd and Vandenberghe, 2004], to solve a concave optimization problem, we can first transform it into a convex optimization problem which has the same constraints but minimizes the negative objective function. Then by solving the transformed convex problem, we obtain an optimal solution that also maximizes the original concave problem.

By transforming the MaxEnt problem, we obtain the following optimization problem:

$$\min_{P\in\mathcal{P}} g'_0 = -g_0 = \mathbb{E}_{D\sim P,\mathbb{D}}[\log P(D)], \tag{3.2.4}$$

subject to the same constraints (3.2.2) and (3.2.3). Since the objective function g_0 of the MaxEnt problem is concave on domain $\mathcal{P}$, hence its negative $g'_0 = -g_0$ is convex on the same domain. And as stated before the constraints (3.2.2) and (3.2.3) are also convex, our transformed optimization problem is indeed a convex problem. It has a global minimum solution that simultaneously solves our MaxEnt problem (3.2.1 - 3.2.3). Hence this minimum solution also can be denoted as $P^*_{\mathcal{B}}$.

To solve the convex optimization problem, we opt to use the *Lagrange method*. We form a *Lagrangian* $L : \mathcal{P} \times \mathbb{R}^{|C|} \times \mathbb{R} \to \mathbb{R}$ by augmenting the convex objective function (3.2.4) with a weighted sum of the constraint functions (3.2.2) and (3.2.3):

$$L(P(\cdot), \boldsymbol{\lambda}, \mu) = g'_0(P(\cdot)) + \sum_{c\in C} \lambda_c g_c(P(\cdot)) + \mu h(P(\cdot)) \tag{3.2.5}$$

where $\boldsymbol{\lambda}$ is a vector of *Lagrange multipliers*, and each vector component λ_c is associated with a constraint c of form (3.2.2), μ is the multiplier associated with constraint (3.2.3). Based on the Lagrangian (3.2.5), the convex problem (3.2.4) can be equivalently expressed (see A.2) by the *primal* optimization problem:

$$\min_{P(\cdot)} \left[\max_{\boldsymbol{\lambda},\mu} L(P(\cdot), \boldsymbol{\lambda}, \mu)\right] \tag{3.2.6}$$

Because of the equivalence, the optimal solution of the primal problem is also the solution of the convex problem (3.2.4), i.e., $P^*_{\mathcal{B}}$. We denote the optimal value of the objective function in above primal problem as p^*.

Solving the primal problem requires to first maximize the Lagrangian with respect to the multipliers and then minimize the resulting maximum with respect to the distribution function. The maximization of the Lagrangian results in two cases:

$$\max_{\boldsymbol{\lambda},\mu} L(P(\cdot),\boldsymbol{\lambda},\mu) = g'_0(P(\cdot)) + \begin{cases} 0 & \text{if } P(\cdot) \text{ is feasible, } \forall \boldsymbol{\lambda},\mu \\ \infty & \text{if } P(\cdot) \text{ is infeasible, } \exists|\lambda_i| = \infty \text{ or } |\mu| = \infty \end{cases}$$

Hence, in order to minimize (w.r.t. $P(\cdot)$) a non-trivial resulting maximum, one need to search in a set of distributions that satisfy constraints (3.2.2) and (3.2.3), i.e., $\mathcal{P}_C$. However, since it is hard to constructively define the feasible distribution set $\mathcal{P}_C$, directly solving the primal problem becomes impractical. Fortunately, we can still solve our convex optimization problem by solving the corresponding *dual problem*. The dual problem associated with the Lagrangian (3.2.5) reads:

$$\max_{\boldsymbol{\lambda},\mu} \Big[\min_{P(\cdot)} L(P(\cdot),\boldsymbol{\lambda},\mu)\Big]. \tag{3.2.7}$$

As we only have equality constraints for the convex objective (3.2.4), according to Boyd and Vandenberghe [2004, p. 216], in the above dual problem there are no constraints on the multipliers $\boldsymbol{\lambda}$ and μ. This means we can search the optimal value of the multipliers within their domain ($\mathbb{R}^{|C|}$ and $\mathbb{R}$ respectively) to maximize the resulting minimum. Note the above dual problem is concave in terms of the Lagrange multipliers $\boldsymbol{\lambda}$ and μ (see A.3), hence it has a global maximum. We denote the optimal objective function value in above dual problem as d^*. Since our transformed optimization problem (3.2.4) is convex, and it does not contain any inequality constraint, hence for this problem the *strong duality* holds [Boyd and Vandenberghe, 2004]. The strong duality states the optimal values of the primal problem and the corresponding dual problem are equal, i.e.,

$$p^* = g'_0(P^*_{\mathcal{B}}(\cdot)) = d^* \tag{3.2.8}$$

This allow us to find the optimum of the original convex optimization problem (3.2.4) by solving the dual problem (3.2.7).

To solve the dual problem, first notice that the Lagrangian (3.2.5) is a convex functional with respect to distribution function $P(\cdot)$ (see A.4). Hence by equating the derivative of $L(P(\cdot),\boldsymbol{\lambda},\mu)$ with respect to $P(\cdot)$ to zero, we obtain the distribution function that maximizes the Lagrangian (see A.5):

$$\begin{aligned} P(\cdot) &= \exp\Big(\sum_{c\in C} \lambda_c f_c(s_{x_c},\cdot) + \mu - 1\Big) \\ &= \frac{1}{1-\mu}\exp\Big(\sum_{c\in C} \lambda_c f_c(s_{x_c},\cdot)\Big) \\ &= \frac{1}{Z}\exp\Big(\sum_{c\in C} \lambda_c f_c(s_{x_c},\cdot)\Big). \end{aligned} \tag{3.2.9}$$

According to the constraint (3.2.3), a feasible background distributions $P(\cdot)$ is normalized over the data space, so the factor Z in (3.2.9) is essentially a partition function with form:

$$Z(\boldsymbol{\lambda}) = \sum_{D\in\mathbb{D}} \exp\Big(\sum_{c\in C} \lambda_c f_c(s_{x_c}, D)\Big). \tag{3.2.10}$$

Now the distribution function (3.2.9) maximizing the Lagrangian is parametrized only by the Lagrange multipliers $\boldsymbol{\lambda}$. Subject the distribution function with partition function (3.2.10) back to the dual problem (3.2.7), we obtain (see derivation in Appendix A.6):

$$\max_{\boldsymbol{\lambda}} d(\boldsymbol{\lambda}) = -\log(Z(\boldsymbol{\lambda})) + \sum_{c\in C} \lambda_c f_c(s_{x_c}, D_{\text{obs}}). \tag{3.2.11}$$

As the objective function $d(\boldsymbol{\lambda})$ is smooth and concave with respect to $\boldsymbol{\lambda}$ (see A.7), the above problem is a unconstrained concave optimization problem. It has a global optimum solution which can be obtained by applying standard numerical techniques, e.g., Gradient Descent Method [Boyd and Vandenberghe, 2004, p. 466]. We denote this optimal solution as $\boldsymbol{\lambda}^*$.

Finally, we obtain the solution $P^*_{\mathcal{B}}$ of the MaxEnt optimization problem (3.2.1 - 3.2.3) by subjecting the solved multipliers $\boldsymbol{\lambda}^*$ back to (3.2.12). That says, the distribution $P^*_{\mathcal{B}}$ is the feasible background distribution with maximum entropy over data space $\mathbb{D}$:

$$P^*_{\mathcal{B}}(\cdot) = \frac{\exp\left(\sum_{c_\in C} \lambda^*_c f_c(s_{x_c}, \cdot)\right)}{\sum_{D\in\mathbb{D}} \exp\left(\sum_{c\in C} \lambda^*_c f_c(s_{x_c}, D)\right)} \tag{3.2.12}$$

3.3 Computing Subjective Unexpectedness of Pattern Measurements

Since the knowledge of a dataset is essentially conveyed by pattern measurements to a user, it is necessary to quantify the user's subjective unexpectedness with respect to pattern measurements. As there is no previous work has been done in modeling the subjective unexpectedness for pattern measurements, we need to extend the model described in the previous section. In this thesis, we define the subjective unexpectedness of a user observed measurement m_{obs} as the difference between the user's expected measurement $\hat{m}$ and the observed measurement m_{obs}. With the retrieved background distribution $P^*_{\mathcal{B}}$, the user's expected frequency measurement $\hat{m}$ of the pattern measure f_x can be defined as follow:

$$\begin{aligned}\hat{m}_{f_x} &= \mathbb{E}_{D\sim P^*_{\mathcal{B}},\mathbb{D}}[f_x(s_x, D)] \\ &= \frac{\sum_{D\in\mathbb{D}} \exp\Big(\sum_{c\in C} \lambda_c f_c(s_{x_c}, D)\Big) \cdot f_x(s_x, D)}{\sum_{D\in\mathbb{D}} \exp\Big(\sum_{c\in C} \lambda_c f_c(s_{x_c} D)\Big)}\end{aligned} \tag{3.3.1}$$

There are three ways to quantify the difference between a observed measurement m_{obs} and the corresponding expected measurement $\hat{m}$:

$$\text{Absolute Difference:} \qquad |\hat{m} - m_{\text{obs}}| \tag{3.3.2}$$

$$\text{Absolute Ratio:} \qquad 1 - \min\Big(\frac{\hat{m}}{m_{\text{obs}}}, \frac{m_{\text{obs}}}{\hat{m}}\Big) \tag{3.3.3}$$

$$\text{Double Tail p-Value:} \quad 2\min\big(Pr(X \geq m_{\text{obs}}|H = \hat{m}), Pr(X \leq m_{\text{obs}}|H = \hat{m})\big). \tag{3.3.4}$$

In the thesis, we choose the absolute ratio (3.3.3) to quantify the unexpectedness. Because the absolute difference (3.3.2) is too conservative when the scale of the compared values is small. For example, an association pattern x with observed frequency 0.5 and expected frequency 0.6, another pattern with observed frequency 0.05 and expected frequency 0.06, using the absolute difference, the unexpectedness of the second measurement is smaller than the first pattern's. This case can happen when the second pattern has more complex descriptor, hence the scale of its frequency is small in nature. In other words, when measure the unexpectedness based on frequency measurement, the absolute difference introduces preference to the patterns with simpler descriptors. On the other hand, the absolute ratio doesn't have this issue. For both patterns, the absolute ratio as unexpectedness are the same: 0.833.

We ruled out the p-Value because it is computationally expensive. Conceptually, by treating the expected measurement as the null hypothesis, then the p-Value gives the probability that a random measure under the background distribution will be smaller(larger) than the observed measurement. Hence it indicates the difference of m_{obs} and $\hat{m}$. However, since for the background distribution in for (3.2.12), it is unclear whether there exists a closed form solution for computing the corresponding p-value. Calculating the p-value by definition requires to accumulate the probability by enumeration over space $\mathbb{D}$, which introduces to much overhead.

Formally, the subjective unexpectedness of a measurement is defined as:

$$\text{SubUnexp}(m_{\text{obs}}) = 1 - \min\Big(\frac{\hat{m}}{m_{\text{obs}}}, \frac{m_{\text{obs}}}{\hat{m}}\Big). \tag{3.3.5}$$

for the special case that $m_{\text{obs}} = 0$, we define $\text{SubUnexp}(m_{\text{obs}}) = 0$.

As our approach of computing subjective unexpectedness for pattern measurements involves modeling the user's current knowledge about the underlying working dataset, we refer to it as the *knowledge model*. The knowledge model takes the users learned pattern measurements as inputs, and computes subjective unexpectednesses of the new pattern measurements as outputs. There subjective unexpectedness, as demanded, can serve as the indicators of how much new knowledge contained in the corresponding pattern measurements in additional to a user's current knowledge about the underlying data. Moreover, according to the definition of pattern measurements in (section 2.1), any measure function f that maps a pattern x and corresponding dataset D to a real value is compatible with the knowledge model. For example, theoretically we can compute subjective unexpectedness for the measurements defined in (section 2.1), i.e., the lift measurement $f_{\text{lft}}(s_{x_{\text{ass}}}, D_{\text{obs}})$ and the frequency measurement $f_{\text{freq}}(s_{x_{\text{ass}}}, D_{\text{obs}})$ of an association pattern x_{ass} and the subgroup interestingness $f_{\text{sgd}}(s_{x_{\text{sgd}}}, D_{\text{obs}})$ of a subgroup pattern x_{sgd}.

We summarize the knowledge model in (Algorithm 1). Notice that to update the knowledge model, one needs to solve the dual problem (3.2.11). This requires to compute the partition function $Z(\boldsymbol{\lambda})$ by summing over the data space with cardinality exponential in $|R|$, i.e., $|\mathbb{D}| = |\mathbb{V}^{|R|}|$. Such summation also appears when computing the user's expected measurement (3.3.1). This makes the exact computation of subjective unexpectedness in knowledge model impractical. In the next section, we show that such complexity can be reduced exponentially for a specific type of pattern measurements.

3.4 Computing Subjective Unexpectedness of Frequency Measurements

In this section, we show that the computational complexity of the knowledge model is can be reduced exponentially with respect to frequency measurements.

A user observed frequency measurement $f_{freq}(s_x, D_{\text{obs}})$ can be readily converted into constraint with form (3.1.1) by plugging in the definition of the frequency measure (2.1.1):

$$\begin{aligned}\mathbb{E}_{D\sim P^*_{\mathcal{B}}}[f_{freq}(s_x, D)] &= f_{freq}(s_x, D_{\text{obs}}) \\ &= \frac{|support(s_x, D_{\text{obs}})|}{|R|}\end{aligned} \tag{3.4.1}$$

Based on the definition of frequency constraint (3.4.1), the background distribution can be computed and represented more explicitly. First, we obtain the partition function by substituting the expression of frequency constraint (3.4.1) into (3.2.10):

$$Z(\boldsymbol{\lambda}) = \sum_{D\in\mathbb{D}} \exp\Big(\sum_{c\in C} \lambda_c f_{freq}(s_{x_c}, D)\Big) \tag{3.4.2}$$

$$= \sum_{D\in\mathbb{D}} \exp\Big(\sum_{c\in C} \frac{\sum_{r\in R_D} s_{x_c}(v_r)}{|R_D|}\Big) \tag{3.4.3}$$

$$= \sum_{D\in\mathbb{D}} \prod_{r\in R_D} \exp\Big(\sum_{c\in C} \lambda_c \frac{s_{x_c}(v_r)}{|R|}\Big) \tag{3.4.4}$$

$$= \prod_{r\in R} \sum_{r_D\in\mathbb{D}} \exp\Big(\sum_{c\in C} \lambda_c \frac{s_{x_c}(v_r)}{|R|}\Big) \tag{3.4.5}$$

$$= \prod_{r\in R} \sum_{v\in\mathbb{V}} \exp\Big(\sum_{c\in C} \lambda_c \frac{s_{x_c}(v)}{|R|}\Big). \tag{3.4.6}$$

Algorithm 1 Knowledge Model

Update the Knowledge Model:

Require: User observed pattern measurements $f_c(s_{x_c}, D_{\text{obs}}),\ c \in C$, current dataset D_{obs}, attribute set A, number of rows in a dataset $|R|$.

1: Initialize Lagrange multipliers, i.e., $\lambda_c = 0,\ \forall c \in C$.

2: Compute the gradient of the dual function $d(\boldsymbol{\lambda})$ in (3.2.11):

$$\frac{\partial d(\boldsymbol{\lambda})}{\partial \lambda_c} = -\sum_{D \in \mathbb{D}} \frac{\exp\left(\sum_{c' \in C} \lambda'_c f'_c(s_{x'_c}, D)\right) \cdot f_c(s_{x_c}, D)}{\sum_{D \in \mathbb{D}} \exp\left(\sum_{c' \in C} \lambda'_c f'_c(s_{x'_c}, D)\right)} + f_c(s_{x_c}, D_{\text{obs}})$$

.

3: Update:

$$\lambda_c \leftarrow \lambda_c - \Delta t \cdot \frac{\partial d(\boldsymbol{\lambda})}{\partial \lambda_c}$$

where Δt is a step length determined by line search algorithm (see [Boyd and Vandenberghe, 2004, p. 464]) to decrease the value of the dual function.

4: If the gradient's norm $\|\nabla d(\boldsymbol{\lambda})\|$ is small enough, go to step 5, otherwise go to step 2.

5: **return** The computed multipliers $\boldsymbol{\lambda}'$.

Compute Subjective Unexpectedness of Pattern Measurements:

Require: a new pattern measurement $m_{f_{x'}} = f(s_{x'}, D_{\text{obs}})$, updated Lagrange multipliers $\boldsymbol{\lambda}'$, current dataset D_{obs}, attribute set A, number of rows in a dataset $|R|$.

1: Compute the expected pattern measurement:

$$\hat{m}_{f_{x'}} = \frac{\sum_{D \in \mathbb{D}} \exp\Big(\sum_{c \in C} \lambda'_c f_c(s_{x_c}, D)\Big) \cdot f_{x'}(s_{x'}, D)}{\sum_{D \in \mathbb{D}} \exp\Big(\sum_{c \in C} \lambda'_c f_c(s_{x_c} D)\Big)}$$

.

2: Compute subjective unexpectedness for pattern measurement $m_{f_{x'}}$:

$$\text{SubUnexp}(m_{f_{x'}}) = 1 - \min\Big(\frac{\hat{m}_{f_{x'}}}{m_{f_{x'}}}, \frac{m_{f_{x'}}}{\hat{m}_{f_{x'}}}\Big)$$

3: **return** SubUnexp($m_{f_{x'}}$)

where s_{x_c} is the descriptor within the measure f_c with respect to the constraint c. The equation (3.4.5) is based on the assumption that the data records are mutually independent. Observe that from equation (3.4.4) to equation (3.4.6), the domain of the partition function (sample space of the distribution) has been transformed from the dataset space $\mathbb{D} = \mathbb{V}^{|R|}$ to the value space $\mathbb{V}$. This reduces the computation complexity exponentially in $|R|$.

Furthermore, reformulate the Lagrange dual problem based on the frequency constraints by plugging (3.4.6) and (3.4.1) into (3.2.7), :

$$\begin{aligned} \max_{\boldsymbol{\lambda}}\ d(\boldsymbol{\lambda}) &= -\log(Z(\boldsymbol{\lambda})) + \sum_{c \in C} \lambda_c f_{freq}(s_{c_x}, D_{\text{obs}}) \\ &= -|R| \cdot \log(\sum_{v \in \mathbb{V}} \exp\Big(\frac{1}{|R|} \cdot \sum_{c \in C} \lambda_c s_{x_c}(v)\Big) + \frac{1}{|R|} \cdot \sum_{c \in C} \lambda_c |support(s_{c_x}, D_{\text{obs}})| \end{aligned} \tag{3.4.7}$$

By solving the problem (3.4.7), we obtain the optimal multipliers $\boldsymbol{\lambda}^*$. So, the optimal background distribution

$P^*_{\mathcal{B}} : \mathbb{V} \to [0, 1]$ over the value space $\mathbb{V}$ reads:

$$\begin{aligned} P^*_B(V) &= \frac{1}{Z(\boldsymbol{\lambda}^*)} \exp\Big(\sum_{c \in C} \lambda^*_c s_{x_c}(V)\Big) \\ &= \frac{\exp\Big(\sum_{c \in C} \lambda^*_c s_{x_c}(V)\Big)}{\sum_{v \in \mathbb{V}} \exp\Big(\sum_{c \in C} \lambda^*_c s_{x_c}(v)\Big)} \end{aligned} \tag{3.4.8}$$

where V is the random variable that takes value from $\mathbb{V}$.

With the background distribution $P^*_{\mathcal{B}}$, the expected frequency measurement $\hat{m}_{freq}(s_x, D)$ for a pattern x has form:

$$\begin{aligned} &\hat{m}_{freq}(s_x, D)\mathbb{E}_{V \sim P_{\mathcal{B}}, \mathbb{V}}[f_{freq}(s_x, D)] \\ &\quad = \mathbb{E}_{V \sim P_{\mathcal{B}}, \mathbb{V}}[s_x(V) = 1] \\ &\quad = \frac{\sum_{v \in \mathbb{V}} \exp\Big(\sum_{c \in C} \lambda_c s_{x_c}(v)\Big) \cdot s_x(v)}{\sum_{v \in \mathbb{V}} \exp\Big(\sum_{c \in C} \lambda^*_c s_{x_c}(v)\Big)} \end{aligned} \tag{3.4.9}$$

In appendix (A.8), we give an concrete example which illustrates how to update the knowledge model and compute subjective unexpectedness of frequency measurements based on a toy dataset.

Chapter 4

An Efficient Implementation of Knowledge Model

As the computational complexity of the the knowledge model (section 3.3) is exponential in input dataset size, computing subjective unexpectedness for arbitrary pattern measurements is inefficient. Currently we only implemented the the knowledge model that efficiently computes exact subjective unexpectedness for frequency measurements on categorical dataset. As the partition function (3.2.10) is the most expensive part in the knowledge model computation, in this chapter, we describe an algorithm applied in our implementation that efficiently computes the partition function. This chapter also discusses how the knowledge model is applied in the OCM system.

4.1 Computing the Partition Function $Z(\boldsymbol{\lambda})$

The computation of subjective interestingness involves in calculating the partition function $Z(\boldsymbol{\lambda})$ (within dual function (3.2.11) and the expected measurement (3.3.1)). Since only frequency constraints are considered in the current knowledge model implementation, we can rewrite the partition function (3.2.10) as:

$$Z(\boldsymbol{\lambda}) = \Big(\sum_{v \in \mathbb{V}} \exp\big(\frac{1}{|R|} \sum_{c \in C} \lambda_c s_{x_c}(v) \big) \Big)^{|R|}. \tag{4.1.1}$$

Notice the argument $(\frac{1}{R} \sum_c \lambda_c s_{x_c}(v))$ of the exponential function is the sum of the multipliers that correspond to the constraints satisfied by value v. Hence a naive way of computing the partition function is to enumerate through the value space $\mathbb{V}$ and collect for each value v the exponential expression of the corresponding satisfied constraints, i.e., $\{c | c \in C \wedge s_v(c) = 1\}$. However, as the value space $\mathbb{V} = \times_{a \in A} \mathbb{V}_a$ is a Cartesian product of all attribute value domains, its cardinality is exponential in the number of attributes $|A|$, i.e., $O(k^{|A|})$ where $k = \max_{a \in A} |\mathbb{V}_a|$. In OCM, a working dataset can be large, the exponential complexity of the knowledge leads to an inefficient subjective unexpectedness computation, hence a better implementation of computing the partition function is required.

The efficiency of the partition function computation can be improved by applying the idea of loop inversion. First for the inner summation (over constraint set C) in the expression (4.1.1), generate all length $|C|$ conjunctions (with negation) of constraints $c \in C$, denoted as $\mathcal{C}$. For example for constraint set $C = \{c_1, c_2\}$, the set of constraint conjunctions reads $\mathcal{C} = \{\overline{c_1} \wedge \overline{c_2}, c_1 \wedge \overline{c_2}, \overline{c_1} \wedge c_2, c_1 \wedge c_2\}$. Note that different conjunctions $conj \in \mathcal{C}$ are associated with different inner sums (i.e., $\sum_{c \in C^+_{conj}} \lambda_c s_c(v)$), where C^+_{conj} denotes the set of non-negated constraints in $conj$, e.g, for conjunction $conj = \overline{c_1} \wedge c_2$, the corresponding set $C^+_{conj} = \{c_2\}$. Then, to carry out the outer summation (over the value space $\mathbb{V}$), for each constraint conjunction $conj$, compute the number of satisfying values v (i.e., $|\{v | conj(v) = \wedge_{c \in conj} c(v) = 1\}|$) in the value space $\mathbb{V}$, denoted as $|\mathbb{V}_{conj}|$. Finally, the partition function computation is computed by summing over

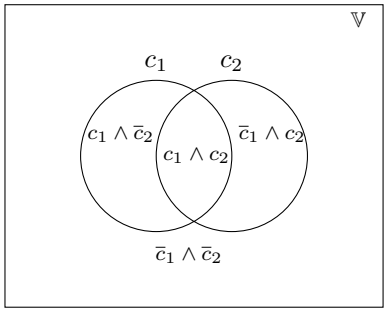
all conjunctions $conj \in \mathcal{C}$ with value $|\mathbb{V}_{conj}| \exp(\frac{1}{|R|} \cdot \sum_{c \in C^+_{conj}} \lambda_c)$, i.e., the product of previously computed satisfying value number $|\mathbb{V}_{conj}|$ and the exponential of the corresponding inner sum $\sum_{c \in C^+_{conj}} \lambda_c$.

The calculation of the inner summation described above is straight forward. Now we describe in more detail how to compute the number $|\mathbb{V}_{conj}|$ of the values satisfying constraint conjunction $conj$. First for all conjunction $conj \in \mathcal{C}$ compute the number of satisfying values $|\mathbb{V}_{conj^+}|$ of corresponding non-negated constraint conjunction $conj^+ = \wedge_{c \in C^+_{conj}} c$. We achieve this by first computing for each attribute domain $\mathbb{V}_a$ the number of satisfactory values, and then multiply them together, i.e., $|\mathbb{V}_{conj^+}| = \prod_{a \in A} |\{v_a | v_a \in \mathbb{V}_a \wedge conj^+(v_a) = 1\}|$. This step has complexity in $O(|A| \cdot |C| \cdot 2^{|C|})$. In the second step, the number of satisfying values $|\mathbb{V}_{conj}|$ is computed by applying the inclusion and exclusion principle proposed by Smith and Gogate [2013]. Notice that the set $\mathcal{C}$ essentially poses a partition on value space $\mathbb{V}$. As an example illustrated in (4.1a), a constraint set $C = \{c_1, c_2\}$ has the corresponding conjunction set $\mathcal{C} = \{\overline{c_1} \wedge \overline{c_2}, c_1 \wedge \overline{c_2}, \overline{c_1} \wedge c_2, c_1 \wedge c_2\}$, which partitions the value space into four disjoint sets. Given a constraint with negation $conj \in \mathcal{C}$, let C_{conj^-} be a set of negated constraints in $conj$, and denote the power set of constraint conjunctions over C_{conj^-} (including empty conjunction) as $2^{C_{conj^-}}$, then the cardinality of satisfactory value set ($|\mathbb{V}_{conj}|$) of conjunction $conj$ is computed as follow:

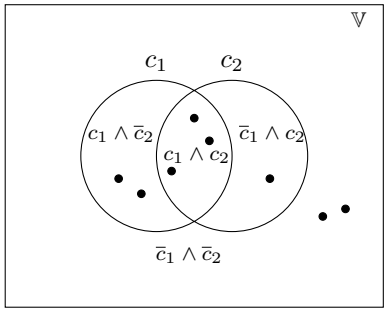
$$|\mathbb{V}_{conj}| = \sum_{l \in 2^{C_{conj^-}}} (-1)^{|l|} |\mathbb{V}_{conj^+ \wedge l}| \tag{4.1.2}$$

where $conj^+ \wedge l$ is a conjunction of non-negated constraints, hence quantity $|\mathbb{V}_{conj^+ \wedge l}|$ is already computed in step one. To illustrate the computation using rule (4.1.2), consider example in (4.1b). The cardinality associated with conjunction $\overline{c_1} \wedge c_2$ can be computed by first adding (inclusion) cardinality of c_2 with value $|\mathbb{V}_{c_2}| = 4$ and then subtracting (exclusion) the cardinality of $c_1 \wedge c_2$ with value $|\mathbb{V}_{c_1 \wedge c_2}| = 3$, i.e., $|\mathbb{V}_{\overline{c_1} \wedge c_2}| = |\mathbb{V}_{c_2}| - |\mathbb{V}_{c_1 \wedge c_2}| = 1$. The complexity of the second step depends only on the number of constraints, hence it is $O(2^{2|C|})$.

Combine the analysis together, the two step computation of the partition function has time complexity of $O(|A| \cdot |C| \cdot 2^{|C|}) + O(2^{2|C|})$, which is essentially in $O(2^{2|C|})$. It is exponential in number of constraint $|C|$, which is also the number of association patterns that a user studied. Compare to the naive method with complexity $O(k^{|A|})$ proposed at the beginning of this section, our two step computation of the partition function is more efficient when the number of patterns studied by a user is small.



(a) Constraint combination set $\mathbb{P}_C$ poses a partition on value space $\mathbb{V}$



(b) Constraint combinations with associated satisfying values (black dots).

We summarize the algorithm of computing the partition function in (Algorithm. 2).

4.2 Applying the Knowlege Model in OCM

We implemented the knowledge model for frequency measurements in the OCM system. This section discusses two main considerations involved in the implementation: a. how to carry out the knowledge model computation in OCM. b. where to use the computed subjective unexpectedness.

Algorithm 2 Partition Function Computation

Require: Lagrange multipliers $\boldsymbol{\lambda}$, constraint set C, attribute set A, number of rows in a dataset $|R|$.

1: generate all length $|C|$ conjunctions of constraints $c \in C$ with negation, denotes as $\mathcal{C}$:

2: **for all** $conj \in \mathcal{C}$ **do**

3: compute the number of values $v \in \mathbb{V}$ satisfying $conj^+ = \wedge_{c \in C^+_{conj}} c$:

$$|\mathbb{V}_{conj^+}| = \prod_{a \in A} |\{v | v \in \mathbb{V}_a \wedge conj^+(v) = 1\}|$$

4: **for all** $conj \in \mathcal{C}$ **do**

5: compute the number of values $v \in \mathbb{V}$ satisfy $conj \in \mathcal{C}$

$$|\mathbb{V}_{conj}| = \sum_{l \in 2^{C_{conj^-}}} (-1)^{|l|} |V_{conj^+ \wedge l}|$$

6: compute the partition function:

$$Z(\boldsymbol{\lambda}) = \Big(\sum_{conj \in \mathcal{C}} |\mathbb{V}_{conj}| \cdot \exp\Big(\frac{1}{|R|} \sum_{c \in C^+_{conj}} \lambda_c \Big) \Big)^{|R|}$$

7: **return** $Z(\lambda)$

4.2.1 Computing Subjective Unexpectedness

The OCM system is operated based on discovery processes. A discovery process consists of three phases. First, the system operates the mining algorithms to discover new patterns and then rank these patterns according to the inferred user preference model. The ranked patterns are presented to users at the end of this phase. Second, while the user is analyzing the patterns (i.e., save interesting ones, delete non-interesting ones), OCM translates the user's actions into her feedback. Third, when the user is done analyzing, the user preference model is updated according to the collected feedback. This updated model is again used to rank the new discovered patterns in the next process. To incorporate the knowledge model into a discovery process, two discovery process phases are modified.

First, compute the subjective unexpectedness in the first phase. In the first phase, new patterns are discovered by the mining algorithms. Hence, the knowledge model needs to compute the subjective unexpectedness for the patterns. As in current implementation of the knowledge model only the frequency measure is considered, we define the subjective unexpectedness of a pattern as the subjective unexpectedness of the pattern's frequency measurement.

Second, update the knowledge model in the third phase. When the user is done with analyzing the result raking, it also indicates that user has studied the patterns the she just saved. The knowledge conveyed by the saved patterns becomes a part of the user's background knowledge, hence the constraint set C as well as the background distribution $P_{\mathcal{B}}$ has to be updated to incorporate the knowledge of conveyed by the pattern measurements within the saved patterns.

4.2.2 Using Subjective Unexpectedness

Our first idea of applying subjective interestingness in OCM is to introduce it as a new feature $\varphi_{\text{SubUnexp}}$ into the feature space $\mathbb{F}$ (see 2.2). In this way, the subjective interestingness of a pattern x will be incorporated into the pattern's utility $u(\varphi(x))$, and automatically used in the ranking mechanism. In this setting, however, when the background distribution is updated, the subjective unexpectedness feature values of the previously studied patterns also will change. As the user learns more about the data, the unexpectedness of these

patterns will decrease to zero, hence the utility model will be biased to prefer the pattern with small subjective unexpectedness. This is obviously not a desired behavior.

Our second thought is to use the pattern subjective unexpectednesses explicitly in the ranking mechanism, trading off the pattern utilities. The OCM framework has already been using a subjective quantity for ranking, that is, the pattern utility. A pattern's utility indicates the user's preference about this pattern, hence it quantifies the pattern's *subjective relativeness* to the user's interests, denoted as SubRelate(x). By trading off it with the subjective unexpectedness OCM would be able to provide a rank that not only relates to but also conveys various new knowledge in addition to what a user already knows. Such trade off, namely the *subjective interestingness* is defined as:

$$\text{SubInterest}(x) = \theta \cdot \text{SubRelate}(x) + (1 - \theta) \cdot \text{SubUnexp}(x), \tag{4.2.1}$$

where $\text{SubRelate}(x) = u(\boldsymbol{\varphi}(x))$ and $\theta \in [0, 1]$ is the parameter that trades off the two subjective quantities. With the new raking factor we can modify the ranking criteria (2.2.1) into:

$$r' = r \cup x, \text{where } x = \underset{x \in Q}{\text{argmax}} \min_{x' \in r} \left(\text{SubInterest(x)} \cdot \text{dist}_{\cos}(x, x')\right) \tag{4.2.2}$$

Put together, the new discovery process has the following steps:

1. compute a ranking of the patterns based on the subjective interestingness.
2. when the user interaction phase finishes, interpret the new saved patterns into constraints and add them into the constraint set C.
3. update user's utility model, then update the knowledge model based on the updated constraint set C.

The effect of the implementation described in this section is evaluated by a user study, which is presented in the next chapter.

Chapter 5

User Study

The knowledge model aims to approximate a user's knowledge gained from a dataset. For pattern measurements with the form defined in section (2.1), the knowledge model can evaluate their subjective unexpectednesses with respect to different users and datasets. The higher subjective unexpectedness a measurement has, the more knowledge about the underlying dataset the measurement may convey to a user. Hence we claim that, by applying the knowledge model, a data mining system should potentially be able to provide its users informative mining results more efficiently. To support this claim, we applied the knowledge model in an interactive data mining system (i.e., OCM) according to the discussion in section (4.2). Regarding to this implementation, following hypothesis is proposed based on the previous claim:

> The OCM system with the knowledge model can provide a user additional knowledge (in additional to what she already knows) more efficiently than the system without the knowledge model.

Generally, to evaluate the benefits of an algorithm or a model, the common practice of the data mining community is performing *intrinsic* formal or empirical evaluations. That is, based on either implicit or explicit assumptions of users, certain measure such as pattern utility or computation speed is used as an evaluation indicator. Following this practice, for the implementation of knowledge model in OCM, one can design an intrinsic evaluation by simulating various virtual users who interact with the system, then evaluate the utilities of the results generated by OCM against each virtual user respectively. However, such simulation is essentially an approximation based on simplified theoretical assumptions about a real-world user. It means that the interacting behavior of a partially simulated user might not always be sensible in the context of the real world situation, and from which only a partially informed evaluation result can be obtained.

Therefore, it is more reasonable to evaluate the benefits of the knowledge model holistically in the context of real world situations which involves real users, i.e., the *extrinsic* evaluations such as anecdotal studies (e.g., Wang and Wang [2002], Ohsaki et al. [2007]) and lab studies (e.g., Ke et al. [2009], Li et al. [2012]). In this chapter, we first discuss in (sec. 5.1) how our hypothesis is translated into a user study design. By running new studies that are instantiated form the same design, the hypothesis can be repeatedly tested (repeatable) against an arbitrary high number of participants without consuming valuable resources (scalable), such as the time of supervision personnel. Then we describe how a user study instance is executed (section 5.2). During the study execution time, we gather the measurements about the participants interacting with systems as well as the data generated through human evaluators reviewing the results produced by the study participants, Finally, by processing the gathered data during the study execution time, we conclude in section (5.3) that our hypothesis is justified by the conducted user study.

5.1 Study Design

In order to test our hypothesis of the knowledge model, we *operationalized* the hypothesis by translating its natural language description into the corresponding *study design*. A study design defines a procedure that

consists of all necessary components for evaluating a hypothesis. Our hypothesis about the knowledge model requires to compare two system variants based on the efficiency that each system provides its users the new knowledge about a dataset. According to this interpretation, operationalizing the hypothesis requires to specify the following four components:

1. **system specifications** correspond to the two systems for comparison mentioned in hypothesis. The OCM system with the knowledge model is the target system that we want to evaluate; the OCM system without knowledge model is the control system that serves as a baseline for the hypothesis testing.

2. **task specifications** represent the tasks that participants need to solve with certain system variant. Against these tasks, one can measure the efficiency that participants discover new knowledge using different system variants. Hence a task specification needs to address the following questions:how to identify the knowledge that are new to participants, how to quantify the new knowledge learned by participants as well as how to measure the "efficiency" of certain system.

3. **assignment logic** defines how the tasks are issued to the study participants in order to obtain required measurements.

4. **system performance metric** defines a criteria based on which one can conclude that one system provides users new knowledge "more" efficiently than another.

As the system specifications are straightforward, we discuss the rest components in detail in this section.

5.1.1 Task Specification

In the user study, participants are required to perform *analysis tasks*, which allows us to gather the data that is later processed to assess our hypothesis. In general, human participants need to understand analysis tasks and solve them by operating certain system variants. This requires to provide the systems the inputs (i.e., *datasets*) that they can operate on and communicate the analysis tasks to participants via certain form of *instructions*. In the study, the hypothesis is assessed based on evaluating the task results produced by the participants. Hence the form of the analysis task *results* has to be clearly defined. With the defined form of analysis task results, a participant will also know when she is done with an analysis task. Finally, the analysis task results are evaluated by human evaluators according to certain predefined metrics. To communicate the evaluation purpose as well as define the evaluation metrics, *evaluation schemes* need to be specified. In the following, we discuss these aspects that are carried out in the knowledge model study.

Datasets In order to measure the knowledge that is "new" to a user, we designed our analysis tasks over two random but fixed fictitious datasets. As for a normal dataset, different participants may have different domain knowledge about it. Hence even from the same pattern discovered in this dataset, the new knowledge that different participants gain varies. That means, for different users the scales of the gained additional knowledge are incomparable, which further makes the study results untraceable. By using fictitious datasets, initially all participants will have no prior knowledge about the datasets. This brings their background knowledge of the dataset to the same level, which serves our need. Moreover, to be explicit about the "new" knowledge, we planted different positive associative relations among the attributes into each datasets. In addition, in order to test whether the knowledge model is data independent, we generated two datasets, and accordingly adjusted the instructions of the corresponding study tasks. For the details of the dataset generating process, see appendix (B.1)

The datasets are called "Lakeland" and "The Plain". Both contain 1000 records of the socio-economic status of the fictitious inhabitants in two different fantasy lands. The datasets are described by the same attributes, where each attribute is associated with a random variable:

- X_1 - *Races of inhabitants*, categorical attribute where $X_1 \in \{\text{Griffin}, \text{Diricawl}, \text{Werewolf}\}$.
- X_2 - *Regions of inhabitants*, categorical attribute where $X_2 \in \{\text{east}, \text{west}, \text{north}, \text{east}\}$.
- X_3 - *Annual income in gold coins*, numeric integral attribute where $X_3 \in \{0, 1, 2, \ldots, 1000\}$.

- X_4 - *Annual spending on health care in percentage of income*, numeric continuous attribute where $X_4 \in [0, 1]$.
- X_5 - *Subjective happiness*, categorical $X_5 \in \{happy, unhappy\}$.

The injected positive associations in dataset "Lakeland" are:

- (**Region** = **North** $\wedge$ **Species** = **Diricawl**),
- (**Region** = **East** $\wedge$ **Annual Health Care Spending** = **High**),
- (**Species** = **Griffin** $\wedge$ **Annual Income** = **High**).

The injected positive associations in dataset "The Plain" are:

- (**Annual Income** = **Normal** $\wedge$ **Happiness** = **Happy**),
- (**Annual Income** $\neq$ **Normal** $\wedge$ **Happiness = Unhappy**),
- (**Annual Income** = **High** $\wedge$ **Annual Heath Care Spending** = **High**).

Tasks instruction As the participants have no prior knowledge about the composed fictitious dataset, any patterns discovered in the datasets are new to them. To ensure the new knowledge conveyed to participants are only in terms of the planted associative relations, participants are instructed to gain basic knowledge about a dataset by studying the elementary statistics provided along with the dataset. Then, for the analysis tasks, each participant is guided by the instructions to use the corresponding system to discover a fixed number of patterns that conveys most knowledge additional what she just learned from the column statistics (see Appendix B.2). By evaluating the knowledge contained in these patterns, we obtain the required quantification of the knowledge that is "new" to a participant. The analysis task instructions are adjusted according to the context of the two fictitious datasets.

Result Definition We define the result of an analysis task as the *three* patterns that in a user's opinion convey most knowledge about the underlying dataset (see Screenshot 5.1). The reason that a result consists three patterns is to align with the number of planted pasterns in the dataset, so that in the best case all aspects of the additional knowledge can be potentially discovered within one result. When performing an analysis task, a user is required to maintain her result and submit once she is satisfied with the result or simply get bored.

Evaluation Scheme In order to quantify the new knowledge learned by a participant, the analysis task result generated by this participant is rated by human evaluators. This purpose is delivered to evaluators by *evaluation instructions* that are presented to the evaluators at the beginning of the evaluation session (see Appendix B.3 for details). In order to bring the evaluators to the same prior knowledge level as the participants, the evaluation task instructions ask the evaluator to first gain basic knowledge about a dataset. Then the instruction teaches participants how to use evaluation metrics (will be discussed soon) to evaluate of the results of three other users. The evaluation task instructions is also adjusted according to the context of the two datasets.

According to the instructions, evaluators rate the results according to a evaluation metric that consists of one question and five answer options (see 5.2). The question reads:

> How much addition knowledge about the current dataset do you gain from this highlighted set of patterns, in addition to the elementary statistics of the attributes?

For a analysis result $X_{u'}$ generated by an participant u', the participant u who rates result $X_{u'}$ are allowed to choose one of five rating values $r_u(X_{u'})$: 0 (none), 1 (almost none), 2 (a little), 3 (some), 4 (a lot). Once a participant finishes and submits her task, her ratings will be stored in the database.

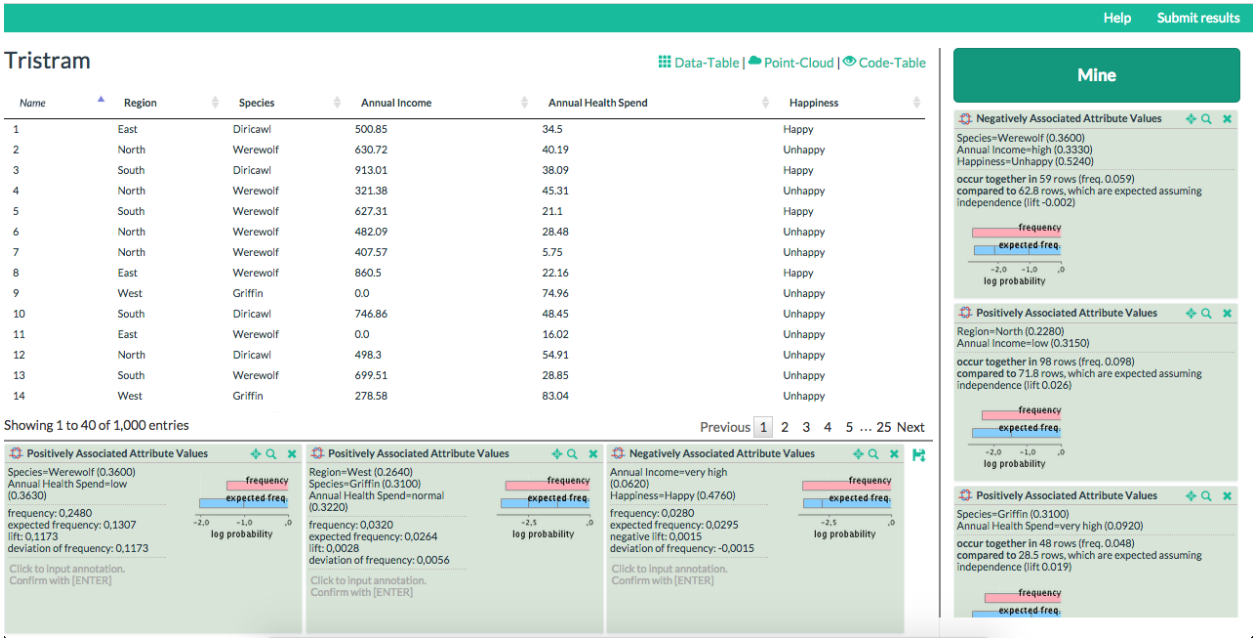


Figure 5.1: Analysis task dashboard in the knowledge model study. The screenshot shows a participant has saved three patterns (left bottom) that might convey to her much knowledge about the underlying data.

5.1.2 System Performance Metric

We quantify the efficiency of a system that provides new knowledge to user according to a *system performance metric*: "average time need by participants to produce a valid result by using a system". This is based on the ratings r_u generated by the human evaluators and the single pattern maximum discovering time t_u that is *automatically* recorded when executing the analysis tasks. The measurement t_u is produced by recording the discovering time $t_{u,x}$ that a user (denoted by u) spend on discovering each result pattern $x \in X_u$, where X_u is the result pattern set submitted by u. Once a participant finishes an analysis task and clicks "Submit Results", the saved patterns as well as the longest time t_{X_u} for discovering one of the three patterns (i.e., $t_{X_u} = \max_{x \in X_u} t_{u,x}$) are submitted to the database.

Therefor, the system performance metric measures the average time for discovering a valid results $\bar{t}_{S,r_{\text{valid}}}$ with system S (i.e., OCM system with (S_{OCMKM})/without (S_{OCM}) the knowledge model), where $r_{\text{valid}} \in 1, 2, 3$ is the "valid average rating" parameter that serves as a threshold. Let $\mathbb{U}_X$ be the set of user who rated analysis result X, denote a set of valid results within an analysis user group $\mathbb{U}$ as $\mathbb{X}_{\mathbb{U},r_{\text{valid}}} = \{X_u | u \in \mathbb{U} \wedge \frac{1}{|\mathbb{U}_{X_u}|} \sum_{u' \in \mathbb{U}_{X_u}} r_{u'}(X_u) \geq r_{\text{valid}}\}$, then we can compute the average time for discovering a set valid result $\mathbb{X}_{\mathbb{U},r_{\text{valid}}}$ with respect to analysis task group $U$by:

$$\bar{t}_{\mathbb{U},r_{\text{valid}}} = \frac{1}{|\mathbb{X}_{\mathbb{U},r_{\text{valid}}}|} \sum_{X_u \in \mathbb{X}_{\mathbb{U},r_{\text{valid}}}} t_{X_u}. \tag{5.1.1}$$

According to our study design, one system variant S corresponds two assignment group, e.g., the OCM system with knowledge model is evaluated by group 2A and 2B. Hence, Formally the average time for discovering a valid result of a system S is defined as:

$$\bar{t}_{S,r_{\text{valid}}} = \frac{1}{|S|} \sum_{\mathbb{U} \in S} \bar{t}_{\mathbb{U},r_{\text{valid}}} \tag{5.1.2}$$

We say that our hypothesis of the knowledge model is justified by a study instance, if for all $r_{\text{valid}} \in 1, 2, 3$, we have measure in the study instance with relation: $\bar{t}_{S_{\text{OCMKM}},r_{\text{valid}}} \leq \bar{t}_{S_{\text{OCM}},r_{\text{valid}}}$. That means, on average

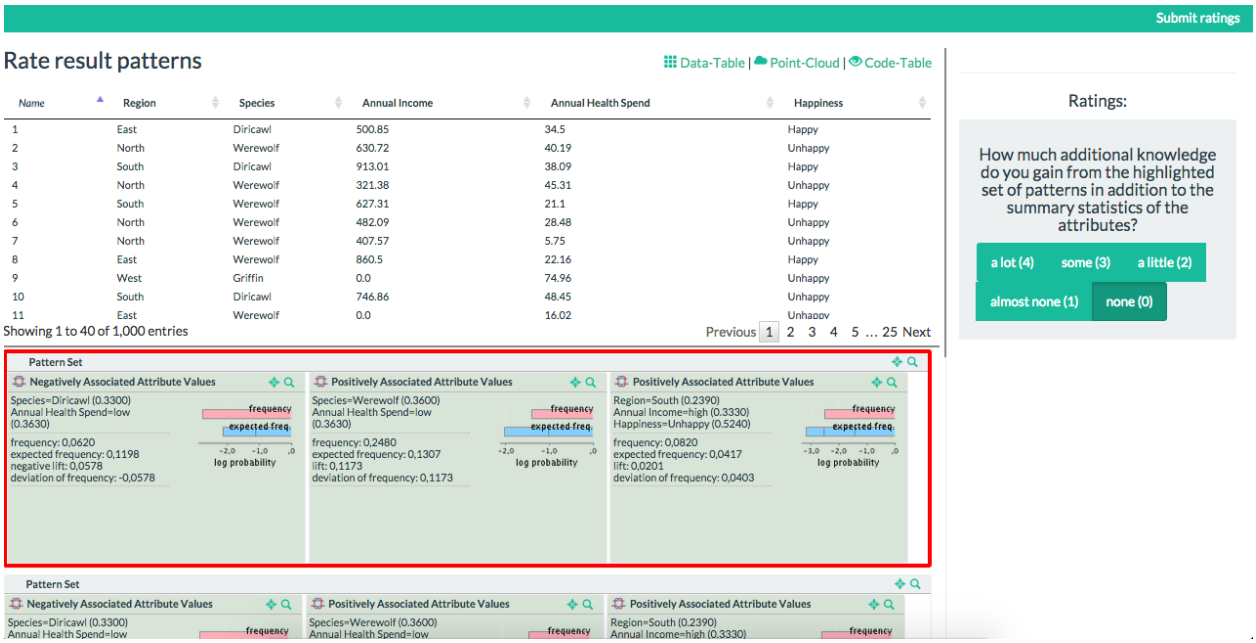


Figure 5.2: Rating task dashboard dashboard in the knowledge model study. The screenshot shows that a participant has selected the first result and given the rating value: 0 (none).

the system with knowledge model spends less time on discovering results that are valid (according to their additional knowledge) than the system without the knowledge model, which aligns with our hypothesis.

5.1.3 Assignment Logic

The assignment logic specifies how analysis tasks are assigned to participants and how the produced analysis task results are distributed to evaluators. The participants will be assigned to perform one of the analysis task according to the *analysis task assignment logic*. Once all participants has finished their analysis tasks, each participant will be required evaluate the analysis task results that are distributed according to the *evaluation assignment logic*.

Analysis Task Assignment Logic. At the beginning of the analysis session, analysis tasks are randomly assigned to participants and each analysis task is performed by relatively the same number of participants. There are two similar analysis tasks corresponding to either one of the datasets (A: Lakeland, B: The Plain). As stated before, the knowledge model study consists of two OCM system variants: target system with the knowledge model (System 1), control system without the knowledge model (System 2). Hence the study participants will be split into four *analysis task groups* according to the different combinations of system variants and datasets (1A, 1B, 2A, 2B). During an analysis session, a participant needs to perform an analysis task according to her group settings, e.g., group 1A uses the OCM system with the knowledge model to perform the analysis task associated with dataset "Lakeland".

Evaluation Assignment Logic. In the evaluation session, the participants who explored the same dataset will cross evaluate three random analysis task results that are related to another dataset, e.g., the participants from group 1A and 2A will cross evaluate the analysis task results generated by group 1B and 2B, and vice versa. This is because the participants performed the same analysis task usually gain different amount knowledge about the data. If their results are evaluated by themselves, the evaluation results will be again provided on participants with different knowledge about the dataset. Hence the evaluation results are biased, and can not serves as evidences for the hypothesis testing. By cross evaluating the analysis task results, the users from group 1A and 2A (1B and 2B) have no prior knowledge about the dataset "The Plain" ("Lakeland"). Such cross evaluation setting allows us to obtain the unbiased ratings for the results from

both analysis tasks, at the same time do not requrie extra participants for evaluation purpose only.

5.2 Study Implementation

Instantiated form on the previously described study design, we conducted a user study to test our hypothesis. We deployed the OCM system a web server, a user can login the system via their own computer. There are in total 16 participants of the study. Each participant receives an email with login account and credential. The user accounts are pre-generated, each account corresponds to one of the OCM system configurations (i.e., with/without the knowledge model). These accounts are evenly and randomly assigned by one of analysis tasks, according to the assignment logic defined in the study design (Section 5.1.3). The user study consists two sessions. During the analysis session, a participant is required to follow the analysis task instructions (Section 5.1.1) that is assigned to her previously and perform the corresponding analysis task. When all participants finish their analysis task, participants are notified by email that the evaluation session is ready. Then participants are asked to follow the evaluation instructions (Section 5.1.1) associated with her task group and evaluate the assigned analysis task results. During the user study, the measurements that are gathered according to the system performance metrics (Section 5.1.2) are collected. These measurements are later analyzed in the study conclusion phase.

5.3 Study Conclusion

With the measurements (w.r.t system performance metrics) collected in the knowledge model study, we computed the average time for discovering a valid result of both the OCM system with/without the knowledge model. The result are summarized in the table (5.3.1). For more insights, see the intermediate measurements of the study listed in appendix (B.4).

System Variant	Avg. Time (s) for Valid Results ($r_{\text{valid}} = 1.0$)	Avg. Time (s) for Valid Results ($r_{\text{valid}} = 2.0$)	Avg. Time (s) for Valid Results ($r_{\text{valid}} = 3.0$)
OCM without KM	694.25	803.96	inf
OCM with KM	**563.63**	**563.63**	**505.5**

Table 5.3.1: Average Time for Valid Results

The table shows that the system knowledge model take less time than the OCM system without the knowledge model to find valid patterns in all different scales of threshold, i.e., $\overline{t}_{S_{\text{OCMKM}},r_{\text{valid}}} \leq \overline{t}_{S_{\text{OCM}},r_{\text{valid}}}$ for all r_{valid}. According to the definition of the system performance metric in (Section 5.1.2), we conclude that this study *justifies* our hypothesis: the OCM system with the knowledge model can provide a user additional knowledge (in additional to what she already knows) more efficiently than the system without the knowledge model.

Chapter 6

Conclusion

This chapter discusses the main results of this thesis. After the discussion, the possible directions of the follow up research are presented.

6.1 Discussion

The goal of this thesis is to develop a mechanism of measuring the amount of the additional knowledge conveyed by mining results in additional to what a user knows. This is motivated by the observation that state-of-art mining systems might present a user mining results that she already knows and filter out the results that are potentially new to the user's knowledge.

To achieve the goal, we first observe that the knowledge of a dataset is essentially conveyed by the pattern measurements together with the pattern descriptions. To model in theory the new knowledge conveyed by pattern measurements, we extended the work of [De Bie, 2013] that originally models the subjective unexpectedness of patterns (see description in section 2.3). Instead, we presented an framework (knowledge model) that quantifies the subjective unexpectedness of pattern measurements in Chapter 3. The knowledge model is applicable in any data mining system that presents its mining results together with general numerical measurements — real values generated by applying measure functions to mining results based on a working dataset. In practice, due to the high computational complexity of the knowledge model, we currently have no efficient implementation that computes the subjective unexpectedness of any numerical measurements. However, for frequency measurements in specific, a better computational efficiency can be achieved (see Section 3.4). The knowledge model for frequency measurements is implemented in the One Click Mining system (see Section 4.2).

To test whether OCM with the knowledge model is able to present users new knowledge more efficiently than OCM without the knowledge model, we developed a user study design (see Section 5.1) that is repeatable for instantiating new studies and scalable for any size of participants group. By conducting a user study that instantiated from the design, we confirmed our hypothesis (see Section 5.3).

6.2 Outlook

Based on the previous discussion of this thesis' contributions, we now turn to the possible research topics that might emerge in the future. There are two major potential research directions:

One direction is the to generalize the knowledge model in theory so that it is compatible with more data types. As the knowledge model described in this thesis is only applicable to categorical datasets, an immediate requirement would be to extend the knowledge model for ordinal and real valued datasets. Moreover, as data is represented in various formats (e.g., graph, stream, etc.), further extending the knowledge model for different data types allows the knowledge model to be applicable to various types of data analysis tasks.

Another potential research direction is to investigate more efficient implementations of the knowledge model. Our current implementation is based on frequency measurements only. This is because the exact computation of the knowledge model in general is exponential in the size of input dataset. However, on one hand, computing the exact value is a very strong requirement. For the cases that do not require the exact computation of subjective unexpectedness, more efficient implementation techniques (e.g., randomized computation) can be applied. To this end, it is important to identify the use cases that do not require exact computation of the knowledge model, and investigate the corresponding efficient implementations. On the other hand, like the knowledge model implementation for frequency measurements, one might also identify other specific types of pattern measurements which allow efficient implementations of the knowledge model.

Bibliography

Mario Boley, Michael Mampaey, Bo Kang, Pavel Tokmakov, and Stefan Wrobel. One click mining — interactive local pattern discovery through implicit preference and performance learning. In *KDD 2013 Workshop on Interactive Data Exploration and Analytics (IDEA)*, 2013.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

Tijl De Bie. Explicit probabilistic models for databases and networks. *CoRR*, abs/0906.5148, 2009.

Tijl De Bie. An information theoretic framework for data mining. In *KDD*, pages 564–572, 2011.

Tijl De Bie. Subjective interestingness in exploratory data mining. In *IDA*, pages 19–31, 2013.

I.M. Gelfand and S.V. Fomin. *Calculus of Variations [by] I.M. Gelfand [and] S.V. Fomin*. Selected Russian publications in the mathematical sciences. Prentice-Hall, 1964.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

Weimao Ke, Cassidy R. Sugimoto, and Javed Mostafa. Dynamicity vs. effectiveness: studying online clustering for scatter/gather. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009*, pages 19–26, 2009. doi: 10.1145/1571941.1571947.

Kleanthis-Nikolaos Kontonasios, Jilles Vreeken, and Tijl De Bie. Maximum entropy models for iteratively identifying subjectively interesting structure in real-valued data. In *ECML/PKDD (2)*, pages 256–271, 2013.

Yunyao Li, Laura Chiticariu, Huahai Yang, Frederick Reiss, and Arnaldo Carreno-Fuentes. Wizie: A best practices guided development environment for information extraction. In *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the System Demonstrations, July 10, 2012, Jeju Island, Korea*, pages 109–114, 2012.

Ingo Mierswa. Rapid miner. *KI*, 23(2):62–63, 2009.

Miho Ohsaki, Hidenao Abe, Shusaku Tsumoto, Hideto Yokoi, and Takahira Yamaguchi. Evaluation of rule interestingness measures in medical knowledge discovery in databases. *Artificial Intelligence in Medicine*, 41(3):177–196, 2007. doi: 10.1016/j.artmed.2007.07.005.

David Brodie Smith and Vibhav Gogate. The inclusion-exclusion rule and its application to the junction tree algorithm. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, 2013.

Shouhong Wang and Hai Wang. Knowledge discovery through self-organizing maps: Data visualization and query processing. *Knowl. Inf. Syst.*, 4(1):31–45, 2002. doi: 10.1007/s10115-002-8192-7.

Appendix A

Maximum Entropy Model

A.1 The MaxEnt Problem is a Concave Optimization Problem

A concave optimization problem consists of a concave objective function, a convex domain, and convex constraint functions [Boyd and Vandenberghe, 2004, p. 137].

First we show that the domain $\mathcal{P}$ of the MaxEnt problem (3.2.1 - 3.2.3) is a convex set. Note that each element $P \in \mathcal{P}$ is a mapping $P : \mathbb{D} \to [0, 1]$. Any two elements P_i and P_j in $\mathcal{P}$, denote their linear combination $P_k = \theta P_i + (1 - \theta)P_j$, where θ is a real number $\theta \in [0, 1]$. Since $P_i(D), P_j(D) \in [0, 1]$, for all $D \in \mathbb{D}$, based on previous construction we have

$$P_k(D) = \theta P_i(D) + (1 - \theta)P_j(D) \in [0, 1], \text{ for all } D \in \mathbb{D}. \tag{A.1.1}$$

This indicates that $P_k \in \mathcal{P}$. According to the definition, the domain $\mathcal{P}$ is convex.

Then we show that the objective function (3.2.1) is a concave objective function over domain $\mathcal{P}$. Explicitly, we have the objective function g_0:

$$g_0(P(\cdot)) = \sum_{D \in \mathbb{D}} -P(D) \log(P(D)) \tag{A.1.2}$$

where $P \in \mathbb{P}$ is a mapping $P : \mathbb{D} \to [0, 1]$. For the expression $-P(D) \log(P(D))$ within the summation operation, the data set D is fixed. By compute its second order derivative with respect to $P(\cdot)$, we have:

$$\frac{\partial^2}{\partial P(D)^2} - P(D) \log(P(D)) = -\frac{1}{P(D)} \tag{A.1.3}$$

by the definition of entropy, $-0 \log(0) = 0$, and $P(D) \in [0, 1]$ the second order derivative is negative for all $P(\cdot)$, hence the expression within the summation operation is concave in terms of $P(\cdot)$. By the property that the non-negative weighted sum of concave functions is concave [Boyd and Vandenberghe, 2004, p. 79], the objective function g_0 is therefore concave.

Finally we prove that the constraint functions g_c, for all $c \in C$ (3.2.2) and h (3.2.3) are convex. Notice that the constraint functions are all of the form $f(\boldsymbol{x}) = A\boldsymbol{x} + b$, where $\boldsymbol{x} \in \mathbb{R}^{|\mathbb{D}| \times 1}$, $A \in \mathbb{R}^{1 \times |\mathbb{D}|}$ and $b \in \mathbb{R}$, they are affine functions. For any $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathbb{R}^{|\mathbb{D}| \times 1}$, $\theta \in [0, 1]$ we have :

$$\begin{aligned} f(\theta \boldsymbol{x}_i + (1 - \theta)\boldsymbol{x}_j) &= A(\theta \boldsymbol{x}_i + (1 - \theta)\boldsymbol{x}_j) + b \\ &= \theta A\boldsymbol{x}_i + (1 - \theta)A\boldsymbol{x}_j + \theta b + (1 - \theta)b \\ &= \theta f(\boldsymbol{x}_i) + (1 - \theta) f(\boldsymbol{x}_j) \end{aligned} \tag{A.1.4}$$

hence by the definition of convex function, the constraints functions $g_c, \forall c \in C$ (3.2.2) and h (3.2.3) are convex functions.

Put all together, we proved that the MaxEnt problem is a concave optimization problem.

A.2 Equivalence of the Lagrange Primal Problem

We proof that the Lagrangian primal problem (3.2.6) is equivalent to the convex optimization problem (3.2.4). The primal problem reads:

$$\min_{P(\cdot)} \Big[\max_{\boldsymbol{\lambda},\mu} g_0'(P(\cdot)) + \sum_{c\in C} \lambda_c g_c(P(\cdot)) + \mu h(P(\cdot))\Big] \tag{A.2.1}$$

First, observe that for any feasible solution $P(\cdot)$ that fulfills constraints (3.2.2 - 3.2.3), we have constraint functions $g_c(P(\cdot)) = 0,$ for all $c \in C$ and $h(P(\cdot)) = 0$. On the other hand, for any infeasible solution $P'(\cdot)$, there are at least one constraint function is non-zero. So, by maximizing the Lagrangian with respect to Lagrange multipliers the function value goes to ∞. We can summarize this formally,

$$\max_{\boldsymbol{\lambda},\mu} L(P(\cdot), \boldsymbol{\lambda}, \mu) = \underbrace{g_0'(P(\cdot))}_{\text{objective of convex problem (3.2.4)}} + \begin{cases} 0 & \text{if } P(\cdot) \text{ is feasible} \\ \infty & \text{if } P(\cdot) \text{ is infeasible} \end{cases}$$

Hence, to find a feasible distribution $P(\cdot)$ that minimizes the primal problem, is equivalent to find a distribution that minimizes function g_0' under constraints (3.2.2 - 3.2.3), i.e., the convex optimization problem (3.2.4).

A.3 Concavity of the Lagrangian w.r.t Multipliers $\boldsymbol{\lambda}$, μ

To see why, we write down the Lagrangian (3.2.5):

$$L(P(\cdot), \boldsymbol{\lambda}, \mu) = g_0'(P(\cdot)) + \sum_{c\in C} \lambda_c g_c(P(\cdot)) + \mu h(P(\cdot)) \tag{A.3.1}$$

It is an affine function in terms of variable λ_c and μ with form $f(x) = Ax + b$ where $x = (\boldsymbol{\lambda}, \mu)^T \in \mathbb{R}^{|C|+1}$, $A = (\lambda_1 g_1(P(\cdot)), \ldots, \lambda_{|C|} g_{|C|}(P(\cdot)), h(P(\cdot))) \in \mathbb{R}^{|C|+1}$, and $b = g_0'(P(\cdot)) \in \mathbb{R}$. As the domain $\mathbb{R}^{|C|+1}$ is convex and any affine function is both concave and convex on a convex domain, hence the Lagrangian is concave with respect to the multipliers.

A.4 Convexity of the Lagrangian w.r.t Distribution Function $P(\cdot)$

To prove the convexity, rewrite the Lagrangian (3.2.5):

$$\begin{aligned} L(P(\cdot), \boldsymbol{\lambda}, \mu) &= \sum_{D\in D} P(D)\log P(D) + \sum_{c\in C} \lambda_c \Big(f_c(s_{x_c}, D_{\text{obs}}) - \sum_{D\in\mathbb{D}} P(D) f_c(s_{x_c}, D)\Big) \\ &\quad + \mu\Big(1 - P(D)\Big) \\ &= \sum_{D\in\mathbb{D}} \Big(P(D)\log P(D) - \sum_{c\in C} P(D) f_c(s_{x_c}, D) - \mu P(D)\Big) \\ &\quad + \sum_{c\in C} f_c(s_{x_c}, D_{\text{obs}}) + \mu \end{aligned} \tag{A.4.1}$$

Only the first summation part in above expression is related to $P(\cdot)$, hence we can replace its inner expression by function:

$$g(P(\cdot)) = P(\cdot)\log P(\cdot) - \sum_{c\in C} P(\cdot) f_c(s_{x_c}, \cdot) - \mu P(\cdot) \tag{A.4.2}$$

as the first part $P(\cdot)\log P(\cdot)$ is convex on domain $\mathcal{P}$ (see A.1.3), and the rest are linear in $P(\cdot)$, hence $g(P(\cdot))$ is a convex function in $P(\cdot)$. As the non-negative weighted of the convex functions is still a convex function, the expression $\sum_{D\in\mathbb{D}} g(P(D))$ is also a convex function in $P(\cdot)$. The Lagrangian in form (A.4.1) consists a convex function in $P(\cdot)$ plus two fix term, so it is a convex functional with respect to the distribution function $P(\cdot)$.

A.5 Derivation of the Background Distribution

The Lagrangian of the optimization problem (3.2.4) reads:

$$\begin{aligned} L(P(\cdot), \boldsymbol{\lambda}, \mu) = &\sum_{D \in \mathbb{D}} P(D) \log P(D) \\ &+ \sum_{c \in C} \lambda_c \Big(f_c(s_{x_c}, D_{\text{obs}}) - \sum_{D \in \mathbb{D}} P(D) f_c(s_{x_c}, D) \Big) + \mu \Big(1 - \sum_{D \in \mathbb{D}} P(D) \Big). \end{aligned} \tag{A.5.1}$$

Denote each part of the above expression as a function:

$$H(D) = \sum_{D \in \mathbb{D}} P(D) \log P(D) \tag{A.5.2}$$

$$G(D) = \sum_{c \in C} \lambda_c \Big(f_c(s_{x_c}, D_{\text{obs}}) - \sum_{D \in \mathbb{D}} P(D) f_c(s_{x_c}, D) \Big) \tag{A.5.3}$$

$$M(D) = \mu \Big(1 - \sum_{D \in \mathbb{D}} P(D) \Big). \tag{A.5.4}$$

compute functional derivative of $H(D)$ with respect to $P(\cdot)$ [Gelfand and Fomin, 1964]:

$$\begin{aligned} \sum_{D \in \mathbb{D}} \frac{\partial H}{\partial P(D)} \phi(D) &= \Big[\frac{\partial}{\partial \epsilon} H\big[P(D) + \epsilon \phi(D)\big] \Big]_{\epsilon=0} \\ &= \Big[\frac{\partial}{\partial \epsilon} \sum_{D \in \mathbb{D}} \big[P(D) + \epsilon \phi(D)\big] \log \big[P(D) + \epsilon \phi(D)\big] \Big]_{\epsilon=0} \\ &= \sum_{D \in \mathbb{D}} \Big[\phi(D) log\big[P(D) + \epsilon \phi(D)\big] + \phi(D) \Big]_{\epsilon=0} \\ &= \sum_{D \in \mathbb{D}} \big[\log P(D) + 1 \big] \phi(D). \end{aligned} \tag{A.5.5}$$

hence the derivative has form:

$$\frac{\partial H}{\partial P(D)} = \log P(D) + 1. \tag{A.5.6}$$

The functional derivative of $G(D)$ with respect to $P(\cdot)$:

$$\begin{aligned} \sum_{D \in \mathbb{D}} \frac{\partial G}{\partial P(D)} \phi(D) &= \Big[\frac{\partial}{\partial \epsilon} G\big[P(D) + \epsilon \phi(D)\big] + \epsilon) \Big]_{\epsilon=0} \\ &= - \sum_{c \in C} \lambda_c \Big[\sum_{D \in \mathbb{D}} \phi(D) f_c(s_{x_c}, D) \Big]_{\epsilon=0} \\ &= - \sum_{D \in \mathbb{D}} \sum_{c \in C} \lambda_c f_c(s_{x_c}, D) \phi(D). \end{aligned} \tag{A.5.7}$$

the derivative has form:

$$\frac{\partial G}{\partial P(D)} = - \sum_{c \in C} \lambda_c f_c(s_{x_c}, D). \tag{A.5.8}$$

The functional derivative of (A.5.4) with respect to $P(\cdot)$:

$$\begin{aligned}\sum_{D\in\mathbb{D}}\frac{\partial M}{\partial P(D)}\phi(D) &= \Big[\frac{\partial}{\partial\epsilon}M\big(P(D)+\epsilon\phi(D)\big)\Big]_{\epsilon=0}\\ &= \Big[-\frac{\partial}{\partial\epsilon}\mu\sum_{D\in\mathbb{D}}[P(D)+\epsilon\phi(D)]+1\Big]_{\epsilon=0}\\ &= -\sum_{D\in\mathbb{D}}\mu\phi(D).\end{aligned} \tag{A.5.9}$$

the derivative has form:

$$\frac{\partial M}{\partial P(D)} = -\mu. \tag{A.5.10}$$

Put together, the derivative of the Lagrangian (A.5.1) with respect distribution function reads:

$$\begin{aligned}\frac{\partial L}{\partial P(D)} &= \frac{\partial H}{\partial P(D)}+\frac{\partial G}{\partial P(D)}+\frac{\partial M}{\partial P(D)}\\ &= 1+\log P(D)-\sum_{c\in C}\lambda_c f_c(s_{x_c},D)-\mu\end{aligned} \tag{A.5.11}$$

equating the derivative (A.5.11) to zero gives the desired distribution:

$$\begin{aligned}\log P(D) &= -1+\sum_{c\in C}\lambda_c f_c(s_{x_c},D)+\mu\\ P(D) &= e^{\mu-1}\cdot\exp(\sum_{c\in C}\lambda_c f_c(s_{x_c},D))\end{aligned} \tag{A.5.12}$$

A.6 Derivation of the Dual Function

The distribution obtained by maximize the Lagrangian with respect to $P(\cdot)$ (3.2.9) reads:

$$[t]P(\cdot) = \frac{1}{Z(\boldsymbol{\lambda})}\exp(\sum_{c\in C}\lambda_c f_c(s_{x_c},\cdot)). \tag{A.6.1}$$

where $Z(\boldsymbol{\lambda})$ is the partition function (3.2.10). Subject the distribution function $P(\cdot)$ back to the Lagrangian yields the dual form of the optimization problem:

$$d(\boldsymbol{\lambda}) = \sum_{D\in\mathbb{D}}P(D)\log P(D)+\sum_{c\in C}\lambda_c\Big(f_c(s_{x_c},D_{\text{obs}})-\sum_{D\in\mathbb{D}}P(D)f_c(s_{x_c},D)\Big)+\mu\Big(1-\sum_{D\in\mathbb{D}}P(D)\Big) \tag{A.6.2}$$

$$= \sum_{D\in\mathbb{D}}P(D)\log\Big(\frac{\exp(\sum_{c\in C}\lambda_c f_c(s_{x_c},D))}{Z(\boldsymbol{\lambda})}\Big)+\sum_{c\in C}\lambda_c\big(f_c(s_{x_c},D_{\text{obs}})-\sum_{D\in\mathbb{D}}P(D)f_c(s_{x_c},D)\big) \tag{A.6.3}$$

$$+\mu\big(1-\sum_{D\in\mathbb{D}}P(D)\big) \tag{A.6.4}$$

$$= \sum_{D\in\mathbb{D}}P(D)\log(\frac{1}{Z(\boldsymbol{\lambda})})+\sum_{D\in\mathbb{D}}P(D)\log\Big(\exp\big(\sum_{c\in C}\lambda_c f_c(s_{x_c},D)\big)\Big) \tag{A.6.5}$$

$$-\sum_{c\in C}\lambda_c\sum_{D\in\mathbb{D}}P(D)f_c(s_{x_c},D)+\sum_{c\in C}\lambda_c f_c(s_{x_c},D_{\text{obs}})+\mu(1-1) \tag{A.6.6}$$

$$= \sum_{D\in\mathbb{D}}P(D)\log(\frac{1}{Z(\boldsymbol{\lambda})})+\sum_{D\in\mathbb{D}}P(D)\sum_{c\in C}\lambda_c f_c(s_{x_c},D)-\sum_{c\in C}\lambda_c\sum_{D\in\mathbb{D}}P(D)f_c(s_{x_c},D) \tag{A.6.7}$$

$$+\sum_{c\in C}\lambda_c f_c(s_{x_c},D_{\text{obs}})+\mu(1-1) \tag{A.6.8}$$

$$=-\log(Z(\boldsymbol{\lambda}))\sum_{D\in\mathbb{D}}P(D)+\sum_{c\in C}\lambda_c f_c(s_{x_c},D_{\text{obs}}) \tag{A.6.9}$$

$$=-\log(Z(\boldsymbol{\lambda}))+\sum_{c\in C}\lambda_c f_c(s_{x_c},D_{\text{obs}}). \tag{A.6.10}$$

Notice that the partial derivative of the dual function vanishes with respect to its optimum $\lambda^*_{c'}$:

$$\begin{aligned}\frac{\partial L(\boldsymbol{\lambda^*})}{\partial\lambda^*_{c'}}&=-\frac{\partial}{\partial\lambda^*_{c'}}\log\Big(\sum_{D\in\mathbb{D}}\exp\big(\sum_{c\in C}\lambda^*_c f_c(s_{x_c},D)\big)\Big)+\sum_{c\in C}\lambda^*_c f_c(s_{x_c},D_{\text{obs}})\\&=-\sum_{D\in\mathbb{D}}\frac{\exp\big(\sum_{c\in C}\lambda^*_c f_c(s_{x_c},D)\big)\cdot f_{c'}(s_{x_{c'}},D)}{\sum_{D\in\mathbb{D}}\exp\big(\sum_{c\in C}\lambda^*_c f_c(s_{x_c},D)\big)}+f_{c'}(s_{x_{c'}},D_{\text{obs}})\\&=-\sum_{D\in\mathbb{D}}\Big(P(D)f_{c'}(s_{x_{c'}})\Big)+f_{c'}(s_{x_{c'}},D_{\text{obs}})\\&=0\end{aligned} \tag{A.6.11}$$

that says:

$$\mathbb{E}_{D\sim P^*_{\mathcal{B}},\mathbb{D}}[f_{c'}(s_{x_{c'}},D)]=\frac{\sum_{D\in\mathbb{D}}\exp\big(\sum_{c\in C}\lambda^*_c f_c(s_{x_c},D)\big)\cdot f_{c'}(s_{x_{c'}},D)}{\sum_{D\in\mathbb{D}}\exp\big(\sum_{c\in C}\lambda^*_c f_c(s_{x_c},D)\big)}=f_{c'}(s_{x_{c'}},D_{\text{obs}}) \tag{A.6.12}$$

This indicates that by searching for the global maximum of the dual function, we are essentially finding the distribution that has the expectation of a measurement equals to the corresponding measurement in the observed data D_{obs}. By corresponding, we mean that the two measurements generated by the same measure function.

A.7 Concavity of the Lagrange Dual Function Parametrizied Only by $\boldsymbol{\lambda}$

To prove that the Lagrange dual function in from (3.2.11) is concave on its domain $\mathbb{R}^{|C|}$, we rewrite the dual function as:

$$\begin{aligned}d(\boldsymbol{\lambda})&=-\log\Big(\sum_{D\in\mathbb{D}}\exp\big(\sum_{c\in C}\lambda_c f_c(s_{x_c},D)\big)\Big)+\sum_{c\in C}\lambda_c f_c(s_{x_c},D_{\text{obs}})\\&=-\log\Big(\sum_{D\in\mathbb{D}}\prod_{c\in C}\exp\big(\lambda_c f_c(s_{x_c},D)\big)\Big)+\sum_{c\in C}\lambda_c f_c(s_{x_c},D_{\text{obs}})\\&=-\log\Big(\prod_{c\in C}\sum_{D\in\mathbb{D}}\exp\big(\lambda_c f_c(s_{x_c},D)\big)\Big)+\sum_{c\in C}\lambda_c f_c(s_{x_c},D_{\text{obs}})\\&=\underbrace{-\log\Big(\sum_{D\in\mathbb{D}}\exp\big(\sum_{c\in C}\lambda_c f_c(s_{x_c},D)\big)\Big)}_{①}\underbrace{+\sum_{c\in C}\lambda_c f_c(s_{x_c},D_{\text{obs}})}_{②\ Q(\boldsymbol{\lambda})}\end{aligned} \tag{A.7.1}$$

In part ①, the expression $\sum_{c\in C}\lambda_c f_c(s_{x_c},D)$ within the exponential is linear in λ_c, for all $c\in C$. As part ① is a negative logarithm of the sum of exponentials, it is concave on $\mathbb{R}^{|\mathbb{D}|}$ (see proof [Boyd and Vandenberghe, 2004, p. 74]). And as the composition is concave in $\mathbb{R}^{|C|}$ (see proof [Boyd and Vandenberghe, 2004, p. 79]).

The expression in part ② is also concave. By applying the definition of convex function, for $0\leq\theta\leq 1$ and $\boldsymbol{\lambda},\boldsymbol{\lambda}'\in\mathbb{R}^{|C|}$:

att_0	att_1	att_2
a	e	j
a	d	i
b	d	i
b	f	j
c	f	h
c	g	j

Table A.8.1: Toy dataset with three categorical attributes

$$\begin{aligned} Q(\theta\boldsymbol{\lambda} + (1-\theta)\boldsymbol{\lambda}') &= -\sum_{c\in C}\Big(\theta\lambda_c + (1-\theta)\lambda'_c\Big)f_c(D) \\ &= -\sum_{c\in C}\theta\lambda_c f_c(D) - \sum_{c\in C}(1-\theta)\lambda'_c f_c(D) \\ &= -\theta\sum_{c\in C}\lambda_c f_c(D) - (1-\theta)\sum_{c\in C}\lambda'_c f_c(D) \\ &= \theta Q(\boldsymbol{\lambda}) + (1-\theta)Q(\boldsymbol{\lambda}') \end{aligned} \tag{A.7.2}$$

hence the second part is concave in $\mathbb{R}^{|C|}$.

By applying the property that the non-negative weighted sums of concave functions is concave, we can conclude that the dual function is concave.

A.8 An Example of Computing Unexpectedness of Frequency Measurements

Here we give a concrete toy example to show the details of calculating subjective unexpectedness for frequency measurements.

Consider a dataset D given in table (A.8.1), the dataset contains three categorical attributes with value space $\mathbb{V}_{\text{att}_0} = \{a, b, c\}$, $\mathbb{V}_{\text{att}_1} = \{d, e, f, g\}$, and $\mathbb{V}_{\text{att}_2} = \{h, i, j\}$. Consider three association patterns:

- x_1 with $s_{x_1} = \{\text{attr}_0 = a \wedge \text{attr}_1 = e\}$, and has frequency measurement $m_{freq,x_1} = 1/6$,
- x_2 with $s_{x_2} = \{\text{attr}_1 = e \wedge \text{attr}_2 = j\}$, and has frequency measurement $m_{freq,x_2} = 1/6$,
- x_3 with $s_{x_3} = \{\text{attr}_0 = a \wedge \text{attr}_1 = j\}$, and has frequency measurement $m_{freq,x_3} = 1/6$.

At the beginning, the user has not studied any pattern. According to the MaxEnt principle, the background distribution is uniform, hence obtain the expected frequency measurements by counting:

$$\begin{aligned} \hat{m}_{freq,x_1} &= \frac{\sum_{v\in\mathbb{V}} s_{x_1}(v)}{|\mathbb{V}|} = \frac{1}{12} \\ \hat{m}_{freq,x_2} &= \frac{\sum_{v\in\mathbb{V}} s_{x_1}(v)}{|\mathbb{V}|} = \frac{1}{12} \\ \hat{m}_{freq,x_3} &= \frac{\sum_{v\in\mathbb{V}} s_{x_1}(v)}{|\mathbb{V}|} = \frac{1}{9} \end{aligned} \tag{A.8.1}$$

where $\mathbb{V} = \mathbb{V}_{\text{att}_0} \times \mathbb{V}_{\text{att}_1} \times \mathbb{V}_{\text{att}_2}$. Then we compute the subjective unexpectedness for the three pattern

measurements:

$$\begin{aligned}\text{SubUnexp}(m_{freq,x_1}) &= 1 - \frac{1}{2} = 0.5 \\ \text{SubUnexp}(m_{freq,x_2}) &= 1 - \frac{1}{2} = 0.5 \\ \text{SubUnexp}(m_{freq,x_3}) &= 1 - \frac{2}{3} = 0.33\end{aligned} \tag{A.8.2}$$

Now, assume that the user has studied x_1 and x_2 two patterns. Two constraints are added to the constraint set $C = \{c_1, c_2\}$, where c_1 and c_2 are induced by frequency measurements m_{freq,x_1} and m_{freq,x_2} respectively. By solving the optimization problem (3.2.1) we have the Lagrange multipliers: $\lambda_{c_1} = 0.9563$ and $\lambda_{c_2} = 0.9563$. With the updated Lagrange multipliers, the updated expected measurements read:

$$\begin{aligned}\hat{m}_{freq,x_1} &= \frac{\exp(\lambda_{c1}) + \exp(\lambda_{c2}) + \exp(\lambda_{c1+c2})}{2\exp(\lambda_{c1}) + 2\exp(\lambda_{c2}) + \exp(\lambda_{c1} + \lambda_{c2}) + 31} = \frac{1}{6} \\ \hat{m}_{freq,x_2} &= \frac{\exp(\lambda_{c1}) + \exp(\lambda_{c2}) + \exp(\lambda_{c1+c2})}{2\exp(\lambda_{c1}) + 2\exp(\lambda_{c2}) + \exp(\lambda_{c1} + \lambda_{c2}) + 31} = \frac{1}{6} \\ \hat{m}_{freq,x_3} &= \frac{\exp(\lambda_{c1} + \lambda_{c2}) + 3}{2\exp(\lambda_{c1}) + 2\exp(\lambda_{c2}) + \exp(\lambda_{c1} + \lambda_{c2}) + 31} = 0.15149\end{aligned} \tag{A.8.3}$$

the subjective unexpectedness also changes:

$$\begin{aligned}\text{SubUnexp}(m_{freq,x_1}) &= 1 - 1 = 0 \\ \text{SubUnexp}(m_{freq,x_2}) &= 1 - 1 = 0 \\ \text{SubUnexp}(m_{freq,x_3}) &= 1 - 0,91 = 0.09\end{aligned} \tag{A.8.4}$$

as we can see that the subjective unexpectedness of first and second measurement drops to zero, because the the corresponding pattern x_1 and x_2 has already been studied. The unexpectedness of third measurement drops from 0.33 to 0.09 because the semantic knowledge of pattern x_3 can be derived from the first two, hence it become expectable after that the first two have been discovered.

Appendix B

User Study

B.1 Data Generating Process

The datasets used in the knowledge model study are randomly generated subject to the predefined associative relations among the data attributes. These relations are essentially the conditional dependencies among the attribute-associated random variables. These dependencies are represented by the corresponding graphical model in Figure (B.1a) and (B.1b).

(a) graphical model for dataset "Lakeland" **(b)** graphical model for dataset "The Plain"

Figure B.1: Graphical models that represent the dependencies among the attribute within both datasets. X_1 - *Race of inhabitants*, X_2 - *Region of inhabitants*, X_3 - *Annual income*, X_4 - *Annual health care spending*, X_5 - *Happiness*

For dataset "Lakeland", the associated probability distributions are:

$$\begin{aligned}
&X_1 : P(X_1|X_2 = \text{east}) = (0.05, 0.05, 0.9),\ \text{otherwise } P(X_1) = (0.45, 0.45, 0.1)\\
&X_2 : P(X_2) = \mathcal{U}(\{\text{north}, \text{east}, \text{south}, \text{west}\})\\
&X_3 : P(X_3|X_1 = \text{Griffin}) = \mathcal{N}(700, 200),\ \text{otherwise } P(X_3) = \mathcal{N}(500, 200)\\
&X_4 : P(X_4|X_2 = \text{east}) = \mathcal{N}(0.7, 0.2),\ \text{otherwise } P(X_4) = \mathcal{N}(0.5, 0.2)\\
&X_5 : P(X_5) = \mathcal{U}(\text{happy}, \text{unhappy})
\end{aligned} \tag{B.1.1}$$

where $\mathcal{U}$ denotes the uniform distribution, $\mathcal{N}$ denotes the normal distribution. For dataset "The Plain", the associated probability distributions are:

$$\begin{aligned}
&X_1 : P(X_1) = \mathcal{U}(\{\text{Werewolf}, \text{Griffin}, \text{Diricawl}\})\\
&X_2 : P(X_2) = \mathcal{U}(\{\text{north}, \text{east}, \text{south}, \text{west}\})\\
&X_3 : P(X_3) = \mathcal{N}(500, 200)\\
&X_4 : P(X_4|X_3 \geq 750) = \mathcal{N}(0.8, 0.2),\ P(X_4|X_3 < 750) = \mathcal{N}(0.4, 0.2)\\
&X_5 : P(X_5|X_3 \in [350, 750]) = \mathcal{B}(0.7),\ \text{otherwise } P(X_5) = \mathcal{B}(0.2)
\end{aligned} \tag{B.1.2}$$

where $\mathcal{B}$ denotes the Bernoulli distribution.

To generate a dataset, samples are drawn attribute wise according to the previously defined distributions. For dataset "Lakeland" sample sequence of the attributes represented by random variable: $(X_2, X_1, X_3, X_4, X_5)$. For dataset "The Plain" the sequence is: $(X_2, X_1, X_3, X_4, X_5)$. For each attribute, 1000 samples are generated as one column in the corresponding dataset. Hence we obtain a dataset by putting sample columns together, and align the samples by their sampling order.

B.2 Analysis Task Instructions

The analysis task instructions consist of two HTML documents. The first documents describes the analysis task that we designed to test the hypothesis of the knowledge model. The second one is a step-by-step instruction of how to operate the OCM system to solve the analysis task. The instructions are the same for different datasets, except the dataset name is replaced by another one.

B.2.1 Analysis Task Description

As a new employee of the Data Science Department of the Lakeland (The Plain) government, you have to get familiar with the socio-economics status of your country. Go on and use our data mining tool to discover key phenomena from Lakeland (The Plain)'s socio-economic data. The data is a sample of 1000 inhabitants' socio-economic records of in your country.

The data mining tool will propose statements about the data, and measures the associations among the statements. Such information is summarized in graphic representations (patterns) like the figure below:

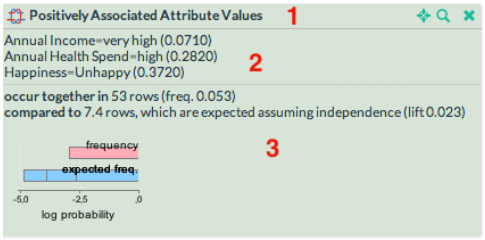


Figure B.2: graphic representation of an association pattern.

- Sec.1 states whether the statements are positively or negatively associated.
- Sec.2 lists the considered statements, along with the frequency (proportion of inhabitants) that each individual statement holds true.
- Sec.3 visualizes the difference between the expected frequency (blue bar) and the actual frequency (red bar) of the statements. The larger the difference the stronger the positive/negative association is.

Patterns can assist you to identify the socio-economic phenomena in the data. For example, according to the statements of the pattern above: the inhabitants who have very high income, high health insurance spending, and also feels unhappy form a group that consists of 5.3% of the sample population. This group population is smaller than the expected group population (7.4%) with the assumption that the statements are independent with each other, which indicates the positive association among the statements. This phenomenon can have many interpretations, one of which can be: "very rich inhabitants in the country are insecure about their healthy, which makes them unhappy". What is your interpretation?

Click on "Next" to continue. On the next page, you will find a more detailed instruction of your task and a quick guide of the mining tool.

B.2.2 System Instruction

Ready for the real tasks? Before that, take some time to go over the step-by-step instructions below of how to operating our mining tool.

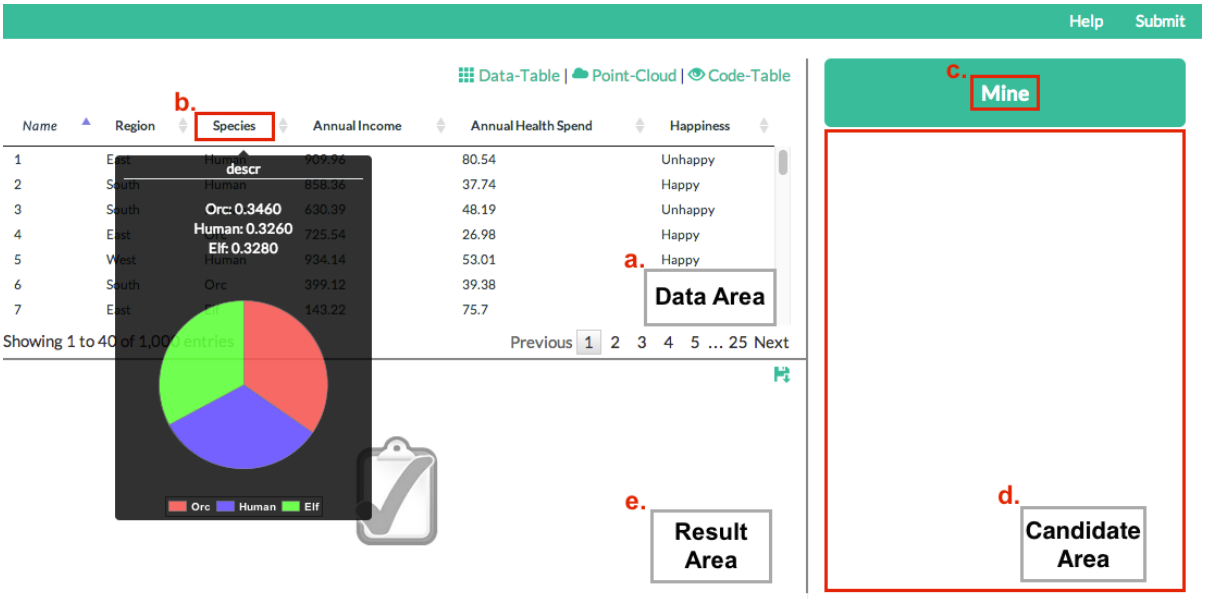


Figure B.3: mining dashboard overview

The dashboard of the mining tool consists of four main components:

- Data Area (Fig. B.3.a) gives you the overview of the data. Within this area, each data column corresponds to a socio-economic attribute. By pointing the cursor at the header of each data column (Fig. B.3.b), you will see the elementary statistics of that column.
- Mine Button (Fig. B.3.c) allows you to request new patterns by clicking on it.
- Candidate Area (Fig. B.3.d) will display the patterns produced by the mining process you just activated.
- Result Area (Fig. B.3.e) allows you to assemble your results by dragging the patterns from the candidate area.

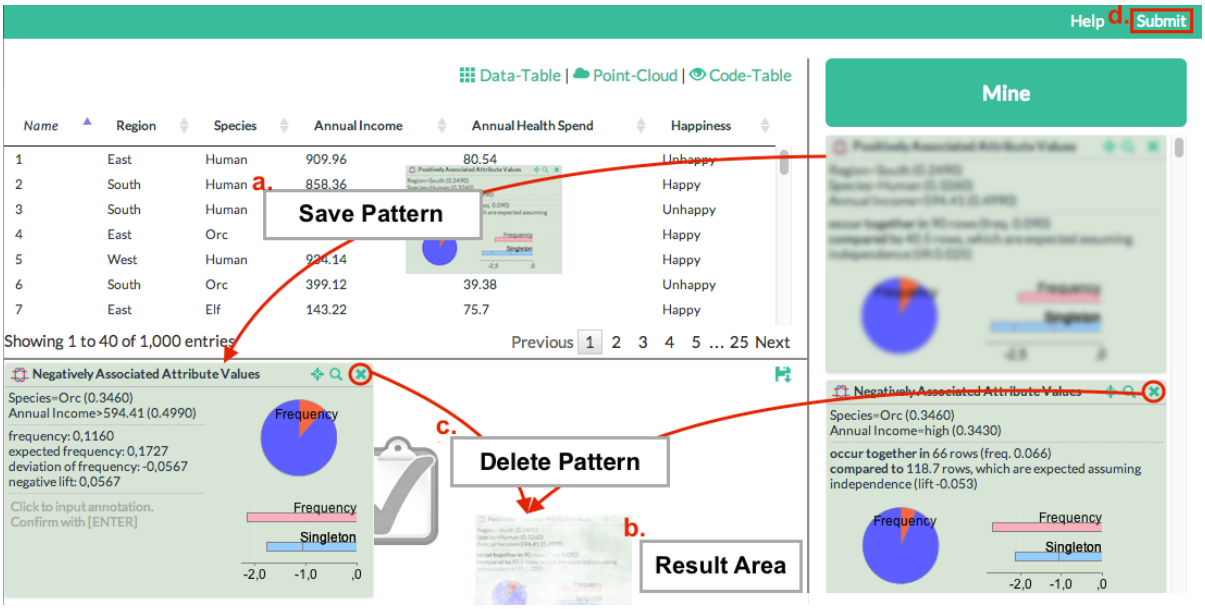


Figure B.4: pattern operations

You need to go through the following steps to perform the analysis task:

I. Study the statistics of each column, and get some elementary knowledge of your country's socio-economic status.

II. Once you are done, click the mine button and patterns will show up in the candidate area. This might take some time, thanks for your patience.

III. Investigate the candidate patterns and save a pattern that you found interesting by dragging it (Fig. B.4.a) to the result area (Fig. B.4.b).

IV. You may delete a non-interesting patterns by clicking the cross icon upper right(Fig. B.4.c).

V. Repeat step II. and IV., refine your pattern selection until you think you have discovered the most interesting three patterns to describe the socio-economic status of Lakeland (The Plain).

When you are satisfied with the results, submit the discovered result by click on the "Submit" button (Fig. B.4.d).

B.3 Rating Task Instruction

As the governor of Lakeland (The Plain), you are requested to evaluate the work of the new employees in the Data Science Department. Please read the following instructions to get familiar with the rating system:

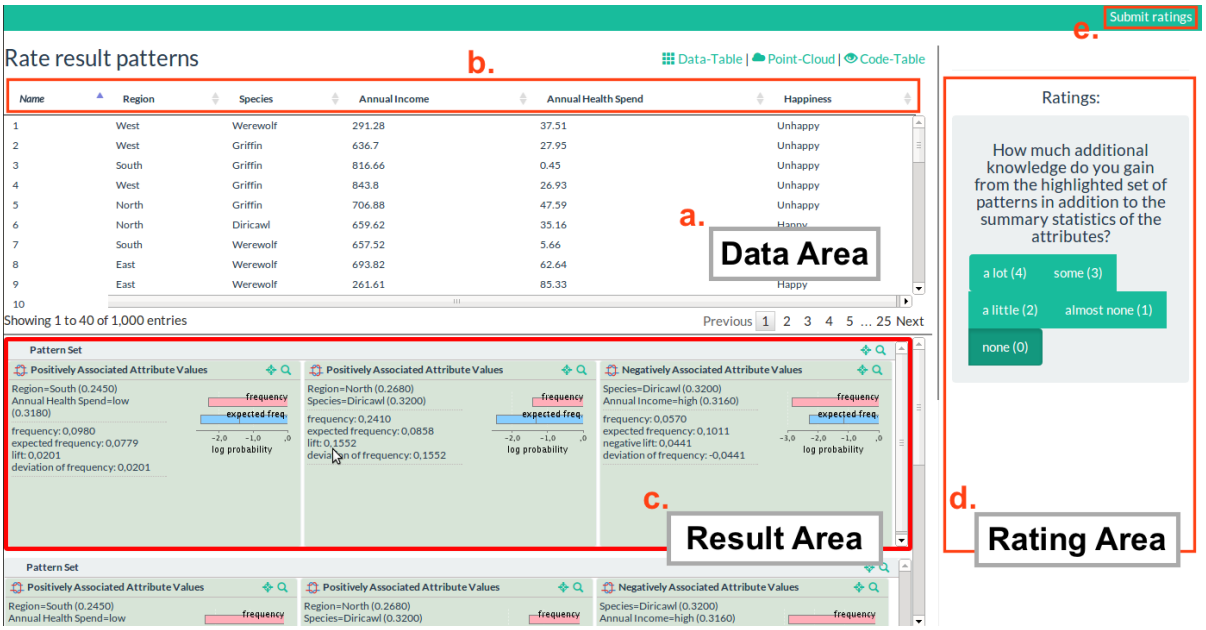


Figure B.5: evaluation dashboard

I. The evaluation dashboard consists of three main components: Data Area (fig. B.5.a) gives you the overview of the data, where each data column corresponds to a socio-economic attribute (fig. B.5.b). By pointing the cursor at the header of each data column, you will see the elementary statistics of that column; Result Area (fig. B.5.c) consists of the results to be rated; Rating Area (fig. d) consists of rating metrics that you can click on to rate the results.

II. Please study the statistics of each column in the data area.

III. Investigate the results discovered by three employees in the result area. Each result consists of three patterns.

IV. To rate a pattern, fist click on it to select (the pattern will be highlighted), then click on the corresponding value of the right-hand-side rating criteria (fig.B.5.d):How much additional knowledge about Lakeland (The Plain) do you gain from the current result, in addition to the elementary statistics of the attributes?

Once you have rated the three results, click "Submit" to save (fig.B.5.e).

B.4 Intermediate Sutdy Measurments

Table B.4.1 summarized the average time of discovering valid results among groups. It show that task group wise the knowledge model take less time than the OCM to find valid patterns in different scales of threshold.

System Variant	Dataset	Avg. Rating (0-4)	Avg. Rating Normalized by Time (s)	Avg. Rating Normalized by User
OCM	Lakeland	2.125	0.00893	1.182
Knowledge Model	Lakeland	**2.500**	0.00812	**1.573**
OCM	The Plain	2.893	0.00530	2.118
Knowledge Model	The Plain	**3.042**	**0.00881**	**2.323**

Table B.4.2: Average Ratings

System Variant	Dataset	Avg. Time (s) for Valid Results ($r_{\text{valid}} = 1.0$)	Avg. Time (s) for Valid Results ($r_{\text{valid}} = 2.0$)	Avg. Time (s) for Valid Results ($r_{\text{valid}} = 3.0$)
OCM	Lakeland	810.25	1029.67	inf
Knowledge Model	Lakeland	**550.75**	**550.75**	**732.0**
OCM	The Plain	578.25	578.25	545.5
Knowledge Model	The Plain	**576.50**	**576.50**	**279.0**

Table B.4.1: Average Time for Valid Results by Group

Apart from the system performance metric $\bar{t}_{S,r_{\text{valid}}}$, by combining the result metrics (i.e., the discovering time $t_{x,u}$ and the user rating $r_u(X_{u'})$ that we can obtain three combined measures:

1. Average rating of an analysis group $\overline{r}_{\mathbb{U}}$. Denote the users of an analysis group as $\mathbb{U}$ (e.g., users of 1A: $\mathbb{U}_{1A}$), let $\mathbb{U}_u$ be the set of users whose analysis task results are evaluated by user u. The average rating of an analysis group $\mathbb{U}$ is defined as:

$$\overline{r}_{\mathbb{U}} = \frac{1}{|\mathbb{U}|} \sum_{u \in \mathbb{U}} \frac{\sum_{u' \in \mathbb{U}_u} (r_u(X_{u'}))}{|\mathbb{U}_u|}. \tag{B.4.1}$$

2. Average rating normalized by time $\|\overline{r}_{\mathbb{U}}\|_{\text{time}}$. This measure is computed as:

$$\|\overline{r}_{\mathbb{U}}\|_{\text{time}} = \frac{1}{|\mathbb{U}|} \sum_{u \in \mathbb{U}} \frac{\sum_{u' \in \mathbb{U}_u} (r_u(X_{u'})/t_{X_u})}{|\mathbb{U}_u|} \tag{B.4.2}$$

3. Average rating normalized by user $\|\overline{r}_{\mathbb{U}}\|_{\text{user}}$. Denote the normalized user rating as $\|r_u(\cdot)\|_{\text{user}} = r_u(\cdot)/\big(\sum_{u' \in \mathbb{U}_u}(r_u X_{u'})\big)$, then we have the average rating:

$$\|\overline{r}_{\mathbb{U}}\|_{\text{user}} = \frac{1}{|\mathbb{U}|} \sum_{u \in \mathbb{U}} \frac{\sum_{u' \in \mathbb{U}_u} (\|r_u(X_{u'})\|_{\text{user}})}{|\mathbb{U}_u|}. \tag{B.4.3}$$

We summarize the corresponding measurements generated by applying the above measures to the knowledge model study in table B.4.2. It shows that the knowledge model setting has average ratings in all aspects better than the system variant without knowledge model. Notice the "Avg. Rating Normalized By Time(s)", for the dataset "Lakeland" OCM is slightly better than the Knowledge model. This is because the knowledge model requires exponential computational time, and it trades off the pattern quality.