Industry and Applications Paper

Predicting "What is Interesting" by Mining Interactive-Data-Analysis Session Logs

Tova Milo
Tel Aviv University
milo@post.tau.ac.il

Chai Ozeri
Tel Aviv University
chaiozeri@mail.tau.ac.il

Amit Somech
Tel Aviv University
amitsome@mail.tau.ac.il

ABSTRACT

Assessing the interestingness of data analysis actions has been the subject of extensive previous work, and a multitude of interestingness measures have been devised, each capturing a different facet of the broad concept. While such measures are a core component in many analysis platforms (e.g., for ranking association rules, recommending visualizations, and query formulation), choosing the most adequate measure for a specific analysis task or an application domain is known to be a difficult task.

In this work we focus on the choice of interestingness measures particularly for Interactive Data Analysis (IDA), where users examine datasets by performing sessions of analysis actions. Our goal is to determine the most suitable interestingness measure that adequately captures the user's current interest *at each step of an interactive analysis session.*

We propose a novel solution that is based on the mining of IDA session logs. First, we perform an offline analysis of the logs, and identify unique characteristics of interestingness in IDA sessions. We then define a classification problem and build a predictive model that can select the best measure for a given a state of a user session. Our experimental evaluation, performed over real-life session logs, demonstrates the sensibility and adequacy of our approach.

1 INTRODUCTION

Assessing the potential *interestingness* of the output generated by a data analysis action has attracted considerable attention both in research and in the industry, and was proven highly useful for tasks such as association rules ranking [18], choosing data visualizations [31], data summaries [6], query formulation [27], etc.

Consequently, a multitude of interestingness measures has been suggested in previous work, each measure attempting to capture a different aspect of the broad "interestingness" concept. For example, *diversity measures* favor data patterns in which elements differ significantly from one another, *peculiarity measures* favor data patterns that display anomalous behavior. Other measures capture *conciseness*, *novelty*, and so on. Consequently, an important (and still open) question is *how can one choose which interestingness measure to employ?* To tackle this exact question, several comprehensive empirical evaluations have been conducted (e.g. [12, 17, 18, 22, 29]). These excellent surveys conclude that (1) there is no single measure that consistently outperforms the rest and (2) the adequacy of specific measures depends heavily on the task at hand and on the application domain.

Whereas most previous work examines the interestingness of specific, singular analysis actions, our work focuses on the interestingness notion within the entire process of Interactive Data Analysis (IDA). There is a growing understanding in the industry and research communities that users analyze datasets *interactively* by issuing a *sequence* of analysis actions of different types (e.g., OLAP, visualizations, mining). Notable and ubiquitous IDA tasks are data exploration, business intelligence (BI), and fraud detection. Typically in IDA, users interact with a dataset by executing a series of analysis actions, referred to as *session.* After issuing an action (e.g. group-and-aggregate, filter, plot, cluster), the user examines its results output (which we call *display*) then decides if and which action to issue next.

Our goal is to predict, at each step of a user's analysis session, what is the most suitable interestingness measure that adequately captures the user's current interest.

If successful, such a predictive model may be highly useful in several analysis "meta" tasks, such as facilitating an evaluation method for analysts' effectiveness, improving existing (and future) analysis recommender systems (e.g. [16, 25, 31]) and enhancing systems for automatic data exploration e.g. ([9, 24]).

To our knowledge, our work is the first to consider dynamically changing interestingness in the context of IDA. Therefore, Our first intent is to demonstrate that interestingness in IDA has different characteristics than the ones assumed in previous work, posing both challenges and opportunities. We identified the following key characteristics, based on an in-depth analysis of real-life session logs:

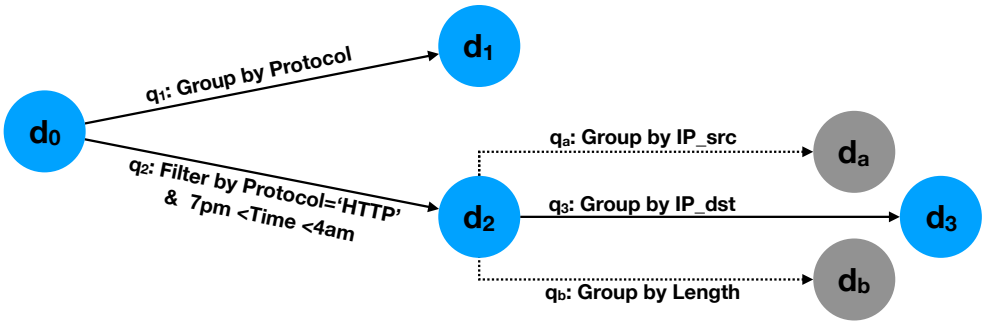
1. There is not one measure that holistically captures "what is interesting" in IDA sessions. When using a single interestingness measure, even if it is the most prevalent one, our experimental evaluation shows that it is inadequate for more than *two thirds* of all our examined cases. Also, many valuable, interesting actions obtain high scores w.r.t. one measure, and low to medium scores by others, hence, different measures need to be employed in different cases.

2. Interestingness (and correspondingly, the measure used to capture it) changes dynamically even in the same user session. We empirically show that within a single user session the "most adequate" interestingness measure changes every 2.2 analysis actions on average.

3. Interestingness is *contextual*. Namely, the analysis context, comprising of previous actions in the same session and their result displays, is, to some extent, correlated with the interestingness preferences of the user (and the measures capturing them).

The following example illustrates the dynamically changing nature of the interestingness notion in a typical IDA scenario.

Example. Clarice is a cyber security analyst assigned to examine inbound network traffic data of a large organization, with the goal of searching for back-door communication channels. She loads the dataset to an IDA interface and performs a sequence of actions, as illustrated in Figure 1 (the sequence of actions is depicted on the upper part of the figure, and the bottom tables depict the actions' results) First, she performs a *group-by* on the field "Protocol" in order to view the amount of traffic of each



d0

Time	IP_src	IP_dst	Protocol	Length
13:00:37	10.1.31.15	8.8.4.4	DNS	76
13:00:41	10.1.31.15	74.25.22.7	TCP	62
...	...	...	...	...
23:56:41	10.2.22.8	8.8.8.8	DNS	72
23:56:41	10.2.22.8	5.8.63.3	SSL	277

d1

Protoco	Count
HTTP	492910
SSL	22479
...	...
SMB	96
NBNS	4

d2

Time	IP_src	IP_dst	Protocol	Length
19:00:00	10.2.26.3	5.8.63.6	HTTP	549.0
19:00:00	10.2.28.12	5.8.63.6	HTTP	350.0
...	...	...	...	...
03:57:42	10.2.23.13	5.8.63.6	HTTP	279.0
03:58:01	10.1.14.11	5.8.63.2	HTTP	471.0

d3

IP_dst	Count
5.8.63.6	100371
5.8.63.2	50538

da

IP_src	Count
10.2.22.5	1624
10.2.28.12	1528
...	...
10.1.15.2	1316
10.1.14.11	1422

db

Length	Count
231	971
238	1966
...	...
723	124
472	631

Figure 1: An Example IDA Session

network protocol. She then returns to the previous display, and issues a *filter* action in order to examine HTTP packets transmitted after business hours (i.e.,between 7pm to 4am). Last, she performs another group-by action on the attribute 'Destination IP', in order to obtain a summary of the uncommon packets (transmitted after business hours) categorized by their destination IP address.

Since Clarice is an expert analyst, assume that all her actions yield interesting result displays. However, *each action is considered interesting according to different measures*: For instance, her first group-by action results in a display (as depicted in Figure 1) that summarizes all traffic according to the network protocol. If we use a *diversity*-based measure (such as Variance [15] or Simpson [15]) to assess the interestingness of this display, it would rank high - as the amount of packets greatly differs between the different protocols. However, if assessed by *peculiarity* based measures (e.g. [19, 28]), which consider displays showing anomalous/extreme patterns as more interesting, this display may obtain a low interestingness score.

In contrast, the results display of the second (filter) action contains rather unusual packets transmitted after business hours (e.g., having a very small length or issued from IP addresses that are uncommon in the dataset). Therefore, this display may be ranked as highly interesting by *peculiarity* measures. However, it may yield lower scores from *diversity* measures (since the attributes of the unusual HTTP packets are rather evenly distributed).

Last, her third (group-by) action results in a compact summary of the unusual HTTP packets, grouped by their destination IP address. This display is considered interesting according to *conciseness* based measures (e.g. [6]) that favor displays conveying a small, human-readable number of rows that summarize a large number of elements. Indeed, all unusual HTTP packets are outgoing merely two different destination addresses. However, this display obtains a low score from *diversity* and *peculiarity* based measures. ■

As illustrated in the example, **although the expert analyst makes actions resulting in interesting displays - each action is supported (i.e., given a high score) by a different interestingness measure, and obtains low to medium scores by others**.

However, attempting to predict the most adequate interestingness measure at each point in an IDA session poses immediate challenges and questions: (1) How can we derive the "ground truth"? Manual labeling may be possible yet time-consuming and costly. (2) Even the simple task of examining a single action and determining which measure finds it more interesting than others is quite difficult, since the different measures capture different facets of interestingness and have different value ranges and distributions.

To overcome these challenges we propose mining analysis session-logs, containing previous analysis actions performed by the same or other users. Given the current user's state in a session, we search the repository to find points in other sessions that are *similar* to the user's state. We analyze what were the relevant measures for these previous sessions and use them to predict the most adequate measure for the current user.

The key contributions of our work are as follows:

1. A simple yet generic data model for interestingness in an IDA environment. Our model is compatible with different types of interactive analysis platforms, from traditional SQL to OLAP and modern web-based interfaces (such as Splunk and Tableau). Our generic model supports a wide range of existing interestingness measures and can be easily extended to support user-defined measures as well.

2. A-posteriori, offline interestingness analysis. The session logs do not provide any information about what parts of the session were interesting and which measures adequately capture it. We therefore devise methods for deriving the most adequate measure at each point in a *past* session by analyzing the interestingness of the *next-action* performed within the same session. By using new techniques for computing *relative interestingness*, we can properly compare the scores of this action (given by different measures) and determine which one best captures its interestingness.

3. Online Interestingness Prediction. Using the results of the above offline analysis, we define a classification problem and build a predictive model that can select the most adequate measure for a current session-state, without knowing its continuation. This model can be used for a dynamic, context-aware selection of interestingness measures in an *ongoing* session, and

Class	Measure	Definition	Reference
Diversity	Variance	$\frac{\sum_{j=1}^{m}(p_j-\bar{q})^2}{m-1}$	[15]
Diversity	Simpson	$\sum_{j=1}^{m} {p_j}^2$	[15]
Dispersion	Schutz	$1-\frac{\sum_{j=1}^{m} p_j-\bar{q}}{2m\bar{q}}$	[15]
Dispersion	MacArthur	$1+\sum_{j=1}^{m}\frac{p_j+\bar{q}}{2}log_2\frac{p_j+\bar{q}}{2}-\frac{log_2 m-\sum_{j=1}^{m} p_j log_2 p_j}{2}$	[15]
Peculiarity	Outlier Score Function	See [19]	[19]
Peculiarity	Deviation	$\delta_{KL}(\{p'_j\}\|\{p_j\})$	[31]
Conciseness	Compaction Gain	$\frac{\|O\|}{m}$	[6]
Conciseness	Log-Length	$1-\frac{min(\log m, c)}{c}$	Following [26]

Table 1: A Partial List of Interestingness Measures

when combined with analysis assistance tools, can aid users in discovering interesting patterns in the data, compose meaningful visualizations, and so on.

4. Experimental evaluation. We evaluated our framework on real-life IDA session logs [1] acquired from over 50 experienced analysts in the domain of cyber security. Our empirical results show that our system can predict, with high accuracy, the correct measure to be used at each point in a user session.

The paper is organized as follows. In Section 2 we describe our data model for IDA interestingness and articulate the problem of interestingness measure prediction. Section 3 describes our framework, comprising the offline interestingness analysis an the online predictive model. Our experiments are detailed in Section 4. Last, we overview related work in Section 5, and conclude in Section 6.

2 BACKGROUND & PROBLEM DEFINITION

We first present a simple yet generic formal model for the IDA process, then describe the different notions of interestingness that we use. Last, we define the problem of *dynamic interestingness measure selection.*

2.1 IDA: Model and Definitions

An IDA session begins when a user loads a dataset, denoted $\mathcal{D}$, to an analysis UI (could be SQL, OLAP or a visualization-driven interface such as Tableau). Then, the user executes a series of *analysis actions* (e.g. SQL queries or visualization actions) $q_1, q_2, \ldots$, examining the obtained results after each one. The results-set of action q_t, executed at step t is called a *display* (representing the results "screen") and denoted by d_t. The preliminary display is d_0, representing the dataset before any action was performed.

IDA session works intuitively like website navigation - at each point the user may invoke an action or backtrack to a previous display and take an alternative navigation path. We thus model an analysis session as an ordered labeled tree[1], denoted S. The nodes represent displays, and the edges outgoing from each node are labeled by the executed action and lead to the resulting display node.

We use S_t to denote the session after step t, namely the *state* in which the user examines the results display d_t, before deciding whether to execute a next action q_{t+1} or conclude the analysis.

[1]If the same display is generated twice (yet on different paths) it is represented by two different nodes

Figure 1 illustrates an *analysis session tree* that corresponds to our running example, in which a user interacts with a dataset of network packets (ignore, for now, the dashed and gray parts). The directed edges represent actions q_1-q_3, and the nodes d_1-d_3 represent their corresponding results displays. The root node, d_0 represents the first display of the dataset before any action was invoked.

Last, we assume throughout this work that past analysis sessions are recorded in a *session log*. We denote by $\mathcal{R}$ a repository of such recorded sessions.[2]

2.2 Interestingness Notions for IDA

A typical interestingness measure, denoted i, takes as input an action q and its results display d, and returns a real number $i(q,d) \in \mathbb{R}$ indicating how interesting are the results (d) of the action q (higher score indicates a more interesting action).[3] For brevity, when d is clear from context, we omit it and simply refer to the *interestingness of action* q by $i(q)$.

While a multitude of different interestingness measures exist (See Section 5 for a discussion), w.l.o.g. we focus our attention in this work to eight common measures from the literature that correspond to four different facets of interestingness, following the categorization in [12] and [15].

The formal definitions of the considered measures (along with a corresponding reference) are provided in Table 1. Next, we intuitively describe each measure, then provide several examples.

Diversity. Diversity measures, e.g. *Simpson* and *Variance* [15] rank higher displays whose elements demonstrate notable differences in values. The definitions of the example measures that we use here are stated in Table 1. The notations are borrowed from [15], assuming an *aggregated* results display: m is the number of groups, v_j is an aggregated value for group j, $p_j = \frac{v_j}{\sum_{k=1}^{m} v_k}$ and $\bar{q} = \frac{1}{m}$. Example below.

Dispersion. In contrast to diversity, dispersion measures e.g. *Schutz* and *MacArthur* [15] favor displays consisting of relatively similar elements.[4]

[2]Analysis sessions may either be recorded by the IDA platform, or, when it does not provide such a service, reconstructed from standard query logs by methods e.g. [32].

[3]Some measures consider more information, such as a reference display or a model of the user's prior belief. While our framework can be naturally extended to support such measures we omit them for the simplicity of presentation.

[4]In some cases the inverse score of a diversity measure can be used to evaluate dispersion, and vice versa.

Class	Measure (i)	Interestingness Scores				Relative Scores	
		$i(q_1)$	$i(q_3)$	$i(q_a)$	$i(q_b)$	$\bar{i}(q_3)$ (RB)	$\bar{i}(q_3)$ (N)
Diversity	Simpson	0.65	0.55	0.15	0.63	1	-0.07
Dispersion	Schutz	0.28	0.83	0.91	0.52	1	1.37
Peculiarity	Outlier Score Function	19.37	1.76	1.02	7.05	1	-0.74
Conciseness	Compaction Grain	51176	75454	39819	25706	2	2.2

Table 2: Interestingness Scores

Peculiarity. A display is peculiar if it presents or contains anomalous patterns. An example is a *Deviation*-based measure [31] that ranks a display higher if it demonstrates a difference from some reference display (e.g. the root display d_0); the p_j notation in the formal definition (Table 1) is the same as above, $\{p_j\}$ denotes the discrete distribution of p_j values, and $\{p'_j\}$ denotes the distribution of the aggregated values in the reference display. $\delta_{KL}(A|B)$ is the Kullback-Leibler divergence distance of the two distributions. Another peculiarity measure is the *Outlier Score Function* [19] (OSF) that focuses on the peculiarity of a single element (i.e, a single tuple, group, or cube cell) within the examined display. The final peculiarity score is simply the *maximum* of the elements' individual scores (See [19] for full details).

Conciseness. Such measures consider the *size* of the display, i.e. the number of elements it contains. Intuitively, displays that convey thousands of rows are difficult to interpret, therefore are considered less interesting. *Log-Length* scores a display proportionally to the log of its size, bounded by a constant c. *Compaction-Gain* (CG) compares the size of the particular display to the number of tuples in the original dataset (denoted O in the formula in Table 1).

In Section 5 we discuss other types of interestingness measures e.g. *surprisingness, actionability* and how they can also be incorporated in our framework.

As the reader can observe, each measure values different properties of the data and may rank a given display differently. The following example illustrates the interestingness evaluation according to the measure types presented above.

Example 2.1. Consider again the IDA session described in Figure 1, and the different interestingness measures described in Table 1. Let us assess the interestingness evaluation of actions q_1 (Group by 'Protocol') compared to q_3 (Group by 'Destination IP'). In Table 2, we report the interestingness scores of q_1 and q_3 according to four different measures, one from each interestingness type (we do not exemplify the calculation of each score, for that we refer the reader to the original papers as depicted in Table 1). Let us examine some of the scores:

1. Diversity, Dispersion: The results of q_1 are considered more interesting than of q_3 as $i(q_1) = 0.65$ and $i(q_3) = 0.55$ (See Table 1). This is due to the larger deviation in the groups' size in d_1 than what appears in d_3 which only contains two groups that are rather even in size. In terms of Dispersion, in which displays with less variations yield higher scores, q_3 is indeed more interesting than q_1 ($i(q_1) = 0.28$ and $i(q_3) = 0.83$)

2. Conciseness: In terms of the Compactness Grain measure, which considers the ratio between the number of tuples to the number of elements (e.g. groups) that covers them, we can see that $i(q_1) = 51,176$ while $i(q_3) = 75,454$. q_3 obtains a higher score than q_1 as its results-display d_3 covers a high number of packets in merely two groups (IP addresses), while d_1 covers all packets with a larger number of groups, one for each network protocol.

Interestingness Measure Prediction. Given a predefined set of interestingness measures $\mathcal{I}$, and a user session state S_t after t steps, our goal is to predict which measure in $\mathcal{I}$ adequately captures "what is interesting" at this point of the session. **Our main hypothesis in this work is that interestingness (and therefore the adequacy of measures) is *contextual*, hence correlated with previous actions taken by the user in the same session.** We illustrate this with our running example.

Example 2.2. Consider the Example Session in Figure 1, at state S_2, i.e. when the user examines Display d_2, before invoking the next action q_3. Our goal, as stated above, is to predict which measure from $\mathcal{I}$ (e.g., Diversity-based, Peculiarity, Conciseness, etc.) best captures the interestingness at this moment. While this seems like a challenging task, examining the previous actions in the session provides some intuition regarding which measure is preferable: In q_2 the user filters all packets to focus on *unusual HTTP traffic occurring after business hours*. As she examines a long list of anomalous elements, it is likely that she is interested in a more *concise* display that summarizes the data, rather than a display that demonstrates another peculiar pattern or one with high Dispersion.

Following this premise, we form a supervised multi-class classification problem that considers session-states as "samples" and assign them a "label" corresponding to interestingness measures in $\mathcal{I}$. In other words, we will assemble a training set containing labeled samples of the form $\langle S_t, i\rangle$ and build a predictive model $F(S_t) \approx i$ that best fits the training data.

To properly define this process, one needs to (1) develop means to determine "what is interesting" in a current state of an IDA session and which measure captures it best. (2) Once this is determined, develop a classification model for predicting what measure best captures the interestingness for the current session state.

We address the two issues in the following section, where we explicitly define the predictive task and model.

3 INTERESTINGNESS PREDICTION FRAMEWORK

We next describe our solution for interestingness measure prediction in IDA sessions. Before we describe our predictive model, we provide (Section 3.1) a set of techniques for *offline interestingness analysis*, in which we retrospectively derive what was the most suitable measure $i \in \mathcal{I}$ for a session state S_t, using the next action q_{t+1}. Then we describe in Section 3.2 how these techniques are used when constructing the training set and building the predictive model.

3.1 Offline Interestingness Analysis

Theoretically speaking, there are several possible ways to examine a session state S_t and determine what is the most suitable measure for it.

First, one can perform *manual labeling* by using expert analysts, familiar with interestingness measures, that can label a session state S_t with the most suitable measure as they see it. The problem with this approach is that it requires, first of all, a great manual effort that is not easily transfered to other contexts (e.g. IDA session logs performed on different datasets or for different purposes). Also, existing measures are often not intuitive even to expert analysts. A second approach could be prompting for user feedback at each session-state, gathering details about the user's intention, goals, and details regarding if and why the current display is interesting (w.r.t. each facet captured by the measures in $\mathcal{I}$). But this method, like the previous one, requires a considerable manual effort (this time by the users performing the sessions).

In contrast to these approaches, we devise means for deriving the adequate measure for a session state S_t solely by examining the continuation of the session. Assume for a moment that our repository only contains actions resulting in interesting displays. Then our key assumption is that for a session state S_t, **if the next action in the same session q_{t+1} is interesting *enough*** (we explain how this is determined in the sequel), **then we can use it to derive what is the measure that best captures the interestingness at S_t**.

A simplistic implementation of the above assumption is to simply choose, given a session state S_t, the measure $i \in \mathcal{I}$ that produces the highest score$i(q_{t+1})$. However, since the measures in $\mathcal{I}$ may produce scores of different value ranges and distributions, this method must be refined.

We therefore devise two interestingness *comparison methods* that are largely impartial to such biases. In each comparison method, we first compute, w.r.t. each measure i, the *relative interestingness score* of an action q denoted $\bar{i}(q)$, which is *comparable* to the relative scores $\overline{i'}(q)$ obtained by other interestingness measures $i' \in \mathcal{I}$. Once we have the unbiased, relative scores, we can simply choose a measure yielding the *maximal* relative interestingness as one that best captures the interestingness of an action q_{t+1}, hence is suitable for S_t. We call a measure *dominant* w.r.t. action q, denoted $i^\star(q)$, if it yields the maximal relative interestingness, i.e. $i^\star(q) = argmax_{i \in \mathcal{I}}(\bar{i}(q))$.

Last, we note that in real-life analysts are imperfect, hence IDA sessions may contain erroneous/redundant actions or simply uninteresting ones. We will show in the sequel how our analysis is used to eliminate such actions and minimize their negative effect.

We start by describing the Reference-Based Comparison method, which is comprehensive but expensive to compute. To overcome this, we then present an alternative method, the Normalized Comparison, which also reduces the score bias but requires a lower computational cost.

Reference-Based Comparison. Our first method for unbiased interestingness comparison, denoted Reference-Based comparison, examines the score of an action q as obtained by a particular measure and compares it to the scores achieved when employing *alternative actions*. Hence, instead of comparing individual scores of different measures for an action q, we first calculate how "high" each measure ranks q compared to a reference set of alternative action, denoted $R(q)$ (in Section 4 we explain how we generate $R(q)$ in our prototype implementation). Then we can simply derive that the measure i which ranks q the highest, is the one that best captures its interestingness (in case there is a tie, all measures that yield the highest relative interestingness are returned).

Algorithm 1 $ReferenceBasedComparison(\mathcal{I}, \langle q, p, d \rangle, R(q))$

```
1: for q' ∈ R(q) do
2:     d' ← The results of action q' on display p
3:     for i ∈ I do
4:         Compute the score i(q', d')
5: for i ∈ I do
6:     Compute the score i(q, d)
7:     ī(q) ← |{q' ∈ R(q) | i(q', d') ≤ i(q, d)}|
8: return argmax_{i∈I}(ī(q))
```

The Reference-Based comparison is depicted in Algorithm 1. It takes as input a set of measure $\mathcal{I}$, a tuple $\langle q, p, d \rangle$ which includes an action q together with its *parent* display p (i.e., the display on which q was employed), and its results display d, and last, a set of *alternative* actions, denoted $R(q)$. It then works as follows.

First, we execute each action in the reference set $R(q)$ (from the same parent display p of action q) then calculate its interestingness scores w.r.t. each measure in $\mathcal{I}$ (Lines 1-4). Next, we calculate the raw scores for q w.r.t. each measure $i \in \mathcal{I}$ (Line 6), then derive the *relative interestingness* score of q , denoted $\bar{i}(q)$, by counting the number of actions in $R(q)$ that obtained a lower interestingness score than q (Line 7). Last, we return the dominant measure(s) $i^\star(q)$ that produced the highest relative interestingness score (Line 8).

Example 3.1. We use the Reference-Based method to assess which is the dominant interestingness measure for action q_3 in our example session (Figure 1). While we can see from Table 2 that d_3 has high Conciseness score, it is also rather disperse. However is it more *disperse* than *concise*, or vice versa?

Let $R(q_3)$ be the set of alternative actions $\{q_a, q_b\}$ (The dashed edges and Grey displays in Figure 1). Their interestingness scores w.r.t. the different measures appear in Table 2. The relative interestingness scores for q_3 appears in the middle section of Table 2. For instance, in terms of Dispersion, since q_3 has a higher score than q_b yet a lower score than q_a its relative score is 1. However, in terms of Conciseness, since the score of q_3 is higher than both q_a and q_b, its relative interestingness (Conciseness) is 2. Indeed, Conciseness yield the highest relative score for q_3, hence is chosen as the dominant measure $i^\star(q_3)$ that best captures the interestingness of action q_3.

While this method completely eliminates the score biases of the different measures, note that it is rather expensive to compute (we demonstrate this in Section 4) as it requires to execute, at each comparison, all alternative actions in the reference set and compute their interestingness scores. Consequently, we devise a second, more efficient comparison method which significantly reduces the score biases using statistical analysis.

Normalized Comparison. The second interestingness comparison method, denoted *Normalized* Comparison, eliminates the score bias due to differences in the range and value distributions, by applying a two-staged normalization process to each measure: (1) to tackle the differences in the measures' value distributions we apply a Box-Cox [5] power transformation that makes the values resemble a normal-distribution. (2) To tackle the differences in the value ranges, we calculate the mean and standard deviation of each measure's (transformed) value distribution then employ z-score standardization [30] so that each computed value now

represents the number of standard deviations the original value differs from the mean. The transformed and standardized score of each action q is defined to be its *relative interestingness score* $\tilde{i}(q)$, and can now be compared to the relative scores produced by other measures in $\mathcal{I}$.

The Normalized Comparison, depicted in Algorithm 2, is divided into two parts.

First, Stage (1) of the normalization process is applied, in a preprocessing manner, to a sample of the score distribution of each measure. The function PreProcess (Lines 1-8) takes as input the set of measures $\mathcal{I}$ and a set QD of actions and their corresponding result displays, i.e. pairs of the form $\langle q, d \rangle$ (such a set can be extracted from the session repository $\mathcal{R}$). It then calculates the interestingness score w.r.t. each measure in $\mathcal{I}$ (Line 4) and transforms its value using the Box-Cox method (Line 5). Last, the mean and standard deviation of the transformed interestingness scores of each measure are returned (Line 8).

Once the preprocessing is done, the Normalized Comparison function (Lines 9-15) can be employed. It takes as input an action q with its results display d, a set $\mathcal{I}$ of interestingness measures and the mean and standard deviation of the transformed scores of each measure, and computes the dominant measure $i^{\star}(q)$ as follows: first, we calculate the interestingness scores $i(q, d)$ w.r.t. each measure in $\mathcal{I}$ and apply the Box-Cox transformation to it (Lines 11-12). Then the mean and standard deviation are updated (Note that Lines 11 to 13 can be skipped if theses values were already computed in the preprocessing phase), and the relative score is calculated by applying the z-score standardization (Line 14). Finally, we return the measure(s) that obtained the highest relative interestingness score $i^{\star}(q)$, as in the Reference-Based method.

Example 3.2. To illustrate the computation, we demonstrate how the Normalized method can be used to assess the measure $i^{\star}(q_3)$ which gives the highest relative score to action q_3 in our running example (depicted in Figure 1).

Assume we performed the preprocessing routine and calculated the scores of all other actions in the log, then performed the transformation and standardization described above. The normalized relative scores for action q_3 are depicted in the right-hand section of Table 2. Consistently with the previous example, the highest normalized score for q_3 is given by the *Conciseness* measure, as its score deviates more than 2.2 standard deviation from the mean conciseness scores.

Also, observe that similar scores of different measures can obtain significantly different results after the standardization process. For instance, the absolute scores of q_3 obtained by the *Dispersion* and *Peculiarity* measures are 0.74 and 0.71 (resp.), however their standardized scores are very different (2.39, −0.2 resp.)

In Section 4 we examine the correlation between the Reference-Based and the Normalized methods and compare their execution times. In what comes next, we describe how these methods are used to construct the training set and the predictive model.

3.2 Online Interestingness Prediction

We developed a predictive kNN-based model for selecting the most suitable measure at a particular session-state.

First we discuss what information is used to describe a session state S_t, then how the training set is constructed and the mechanism of the kNN-based classification model.

Algorithm 2 *NormalizedComparison*

```
1:  function PreProcess(I, QD)
2:      for i ∈ I do
3:          for ⟨q, d⟩ ∈ QD do
4:              Compute the score i(q, d)
5:              ĩ(q, d) ← BOX-COX(i(q, d))
6:      for i ∈ I do
7:          Calculate μ̃_i and σ̃_i, the mean and SD of all ĩ(q, d).
8:      Return μ̃_I, σ̃_I, containing μ̃_i, σ̃_i ∀i ∈ I
9:  function NormalizedComparison(I, ⟨q, d⟩, μ̃_I, σ̃_I)
10:     for i ∈ I do
11:         Compute i(q, d)
12:         ĩ(q, d) ← BOX-COX(i(q, d))
13:         Update μ̃_i and σ̃_i
14:         ĩ(q) ← Z-SCORE(ĩ(q, d), μ̃_i, σ̃_i)
15:     return argmax_{i∈I}(ĩ(q))
```

Describing Session States. Recall that a session state S_t, is the subtree of S containing the first t actions and their result displays.

However, as older actions in the sessions may be of less importance to the classification model, we follow [25] and consider in our predictive model only the n most recent actions and displays, which we call the *n-context* of S_t, denoted c_t. More formally, c_t is defined as the minimal subtree of S that covers the most recent $min(n, 2t + 1)$ elements (i.e., displays and actions) up to step t (inclusive).

As an example, the 3-context at step $t = 2$ in our example session in Figure 1 includes Displays d_0 and d_2 and the action q_2.

In Section 4 we evaluate the predictive performance of the model when using n-contexts of various sizes.

Training Set Construction. Building a training set for a given session repository $\mathcal{R}$ and a set $\mathcal{I}$ of interestingness measures is performed as follows:

(1) Extracting n-contexts from the session repository. For each session state S_t in every session S in the repository we first compute its n-context. As we assume that the sessions in $\mathcal{R}$ are already represented as trees, deriving the n-context for a session state S_t can be done by a DFS-like traversal: Starting from display d_t, we process the nodes (i.e., displays) in reverse to the order of execution of their corresponding actions, considering only actions executed before step t until the size of the induced subtree (nodes+edges) reaches n, which is a configurable parameter in our framework.

For each session state S_t we keep a pair $\langle c_t, q_{t+1} \rangle$ comprising its corresponding n-context and the consecutive action q_{t+1} which will be used to derive its label (i.e., the suitable measure for that session state). For space efficiency, it is sufficient to store for each n-context only pointers to the original actions in the log rather than duplications. In Section 4 we explain how n-contexts are extracted and stored in our implementation.

(2) Assigning labels to n-contexts. For each pair $\langle c_t, q_{t+1} \rangle$, we use either one of the two comparison methods described above to find the dominant measure $i^{\star}(q_{t+1})$. This measure is assigned as a label to the n-context c_t, representing the most suitable measure for the corresponding session state S_t (recall that more than one measure may be qualified).

(3) Omitting globally "non-interesting" samples. Naturally, some of the actions in the session repository may be erroneous or simply non-interesting. Consequently, we want to eliminate

from the training set samples whose consecutive action q_{t+1} is not interesting enough w.r.t. any of the measures in $\mathcal{I}$, hence can not be used for deriving the right measure. This is done by using a configurable threshold for the maximal relative interestingness score of q_{t+1}, denoted θ_I. If the relative interestingness obtained by the dominant measure $i^\star(q_{t+1})$ is lower than θ_I then we discard the sample $\langle c_t, q_{t+1}\rangle$ from the training set.

kNN-Based Classification. Once the training set containing labeled n-contexts is constructed, a classification model can be used, given n-context, to predict the *dominant* measure.

In principle, there are multiple techniques for performing supervised classification, however many of them requires a numeric vector representation for the samples (n-contexts in our case). However, we are not aware of such numeric representation of analysis sessions, yet numerous previous works [3, 11, 13, 25] define a notion of distance/similarity for analysis sessions (or a part thereof). For example the measure suggested in [25] uses tree edit distance to compare two session trees, together with two ground metrics that compare individual actions and displays. Alternatively, in [3] the authors suggest a measure based on local sequence alignment. We harness such a distance notion to form a simple kNN classifier: Given an n-context c_t, we search the training set for its k nearest-neighbors, then employ a majority-vote and return the most common label among the nearest neighbors. Last, it may be that some of the k nearest n-contexts may be too distant from c_t. To avoid the negative effect of such cases on the model's output, we use a distance threshold, denoted θ_δ, which is used to enforce a maximal distance (i.e., a minimal degree of similarity) between the kNN set and the given n-context. If the nearest neighbors retrieved are not similar enough, the model does not yield a prediction.

We intuitively explain this using our running example.

Example 3.3. Assume that our session repository $\mathcal{R}$ comprises only the example session depicted in Figure 1. We first extract n-contexts of e.g. size 3 from $\mathcal{R}$: For each $1 \leq t \leq 3$ we create the 3-context c_t: c_1 contains the single node d_0, c_2 contains d_0, q_1, d_1 and c_3 contains d_0, q_2, d_2. Recall from the previous examples that Compaction Gain measure (Conciseness) is the dominant measure for action q_3, therefore is used as a label for c_3.

Assume we are given another user's session on a different network log, however with the following last action q_u: "filter by protocol='SSL' & 10pm < Time<3am" (which is intuitively similar to q_2 in our example session in Figure 1). After extracting the n-context (containing q_u, its results, and its parent display) from the new session state, we predict what is the adequate measure using the kNN model: If using $k = 1$, then the most similar n-context in the repository is c_3, hence Compaction Gain will be return as prediction.

In Section 4 we explain how we evaluated the predictive model in various settings and compared its performance to several baseline approaches.

4 EXPERIMENTAL EVALUATION

We applied our offline interestingness analysis methods as well as the predictive model on an IDA session log containing real life analysis actions. We begin by describing the session log and our implementation choices, then describe our findings from the offline analysis. Last we evaluate the accuracy of our predictive model compared to other baselines, then test the effect on performance induced by the model's hyper parameters.

REACT-IDA: A repository of real-life IDA sessions. We used the only publicly available (to our knowledge) collection of recorded analysis sessions performed by real users on real-life datasets [1]. The sessions were collected as part of the experimental evaluation of an existing IDA recommender system [25], developed by some of the authors of this work (we discuss this system in more details in Section 5). The repository contains sessions performed by 56 network security analysts, recruited via dedicated forums, security firms, etc. The participating analysts were asked to explore 4 different network-logs datasets using REACT-UI [23], a dedicated, web-based analysis platform with an easy to use interface supporting data filtering, grouping and aggregation. Each dataset contains raw network logs that may reveal a distinct security event, e.g. malware communication hidden in network traffic, hacking activity inside a local network, an IP range/port scan, etc. After completing an analysis sessions, REACT-UI prompts the user to type a short summary of the findings. Sessions corresponding to summaries that successfully reveal the underlying security event are marked as *successful*. The repository contains a total of 454 sessions comprising 2460 distinct analysis actions, out of which 122 sessions (comprising 757 actions) are successful. The REACT-IDA session database in [1] also contains the original datasets used in the analysis, and the means for regenerating the actions and inspect their result displays, so that the each recorded session can be fully reconstructed.

4.1 Offline Analysis Evaluation

We next describe the application of our offline interestingness analysis methods on the REACT-IDA sessions database, accompanied by selected findings.

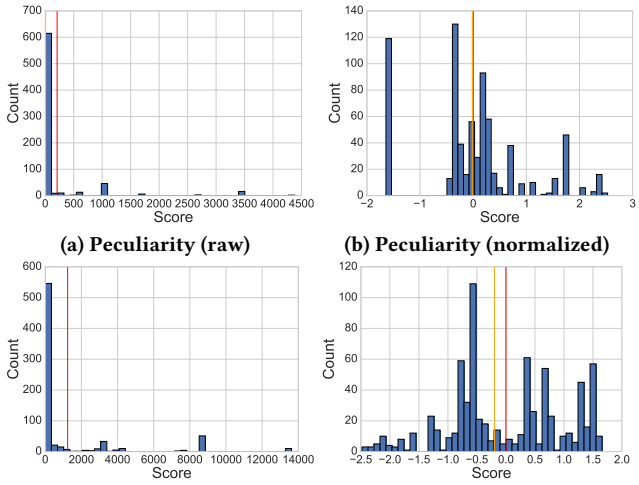
Computing interestingness scores. We re-executed the recorded actions in the REACT-IDA database and computed their interestingness scores w.r.t. all measures presented in table 1. Next, to form an unbiased set of measures $\mathcal{I}$, i.e., that does not contain dependent/highly correlated measures we computed the Pearson Correlation Coefficient for every pair of measures. While the average correlation score was 0.3 we saw that measures from *different types* (e.g. Dispersion, Peculiarity, Conciseness) have an average correlation of 0.071 compared to an average score of 0.543 obtained by measures of the *same type*. Consequently, we experimented with 16 different configuration of $\mathcal{I}$, containing one measure from each type. .

Applying offline comparisons. We applied both comparison methods to the REACT-IDA session database in order to calculate the dominant interestingness measure $i^\star(q)$ for each action q. As for the Reference-Based Comparison, recall that it compares the interestingness scores of an individual action to the interestingness scores of alternative actions. We constructed the reference action set as follows: For each action q with a parent display p we considered all actions in the databases from the same type (e.g. group-by, filter), omitting actions that when executed from display p result in displays comprising less than two rows.

As for the Normalized Comparison, we calculated the statistics over all actions (and their results) in the REACT-IDA database by applying the Box-Cox transformation and z-standardization to their raw interestingness values as described in Section 3.1. Each series of interestingness values (corresponding to a particular measure) was first shifted by a constant in order to eliminate negative scores (this is often required for power transformations). The configurable power parameter λ, which is used as the exponent for the values to be transformed, was determined using

maximum-likelihood estimation, as is standard for such transformations.

As an example, Figure 2 depicts the scores histograms of the Outlier Score Function (Peculiarity) and the Compactness Grain (Conciseness) measures, before and after normalization. The red line in each figure represents the mean score, and the orange line represents the median. While the non-normalized scores are skewed towards zero, we can see that the normalized values distribute much more evenly, resembling a normal distribution.



(a) Peculiarity (raw) **(b) Peculiarity (normalized)**

(c) Conciseness (raw) **(d) Conciseness (normalized)**

Figure 2: Interestingness Scores Histograms

Understanding users' interestingness preferences.

We studied the output of the interestingness comparison methods, namely the dominant measure w.r.t. each recorded action, in order to empirically validate our first two hypotheses presented in the introduction:

(1) Is there one measure/interestingness type that is sufficient to capture "what is interesting" in IDA? If indeed so, then it is sufficient to choose a-priori a single interestingness measure and apply it to all IDA tasks. To answer this question we counted how many actions in the REACT-IDA were labeled with the same dominant measure.

Figure 3 depicts the proportion of actions labeled with measures from the same interestingness type (averaged over all settings of $\mathcal{I}$), when using both the Reference-Based and the Normalized comparison methods.

We can see that in both comparison methods, the most common measure is dominant w.r.t. only 41% of the recorded actions, and the proportions of the rest of the measures types are rather evenly distributed. Due to ties, see that the sum of proportions is slightly larger than 1, i.e. where more than one interestingness measure was found dominant for the same action. Also, note that there are differences in some of the classes' size when a different comparison method is used (mainly in the Peculiarity and Conciseness classes). This is due to slight differences in the comparison base for each method: The Reference-Based method is affected by the parent display of the examined action (from which it was executed), whereas the Normalized is affected by the interestingness scores of other recorded actions, regardless of their parent displays and context.

The above result indicates that *there is no single interestingness measure that can be used for all actions in the repository*. However, as users' IDA sessions are performed on different datasets and for

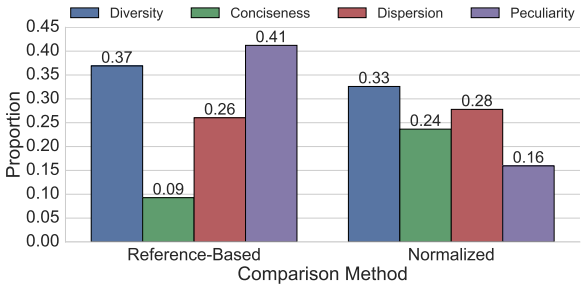


Figure 3: Interestingness Class Labeling Frequency

different purposes, one may still ask whether *one interestingness type is sufficient to capture interestingness within the same session?* If true, it may be sufficient to match a single interestingness measure with certain IDA tasks or datasets rather than dynamically choose a measure for each state in a user session. We therefore examined the relative interestingness scores of actions *of the same session*. We found that *on the course of a single session in the REACT-IDA repository, the dominant measure is changed every* 2.2 *steps* on average. This demonstrates that the interestingness preferences of the user, as well as the measures capturing them, are dynamically changing even within the same IDA session. In Section 4.2 we empirically validate our third hypothesis arguing that *interestingness is contextual* i.e., that one can successfully predict the right measure for a given session state S_t based on its corresponding n-context c_t.

Correlation between the comparison methods. We studied how consistent is the output of the two comparison methods. First, we found that 68% of all recorded actions obtained exactly the same dominant measure as output, by both methods. We then performed a Chi-Square test for independence between the outputs of the comparison methods on all actions: The methods were found highly correlated with a negligible p-value $< 10^{-67}$. The latter result demonstrates the sensibility of our offline interestingness analysis, and that the methods may be used interchangeably.

Execution times.

Applying the offline comparison methods includes three major parts:

(1) Calculating interestingness scores. Recall that the raw interestingness scores are precomputed for each measure in $\mathcal{I}$. While some measures are rather fast to compute (e.g. both Conciseness measures), others (such as the Outlier Score Function) are much more time consuming.

(2) Actions Execution. This part is relevant to the Reference-Based which compares the interestingness scores of a given action to a set of alternative actions. Thus, each such reference action needs to be executed on the same dataset (from the same parent display) by the IDA platform.

(3) Computing relative interestingness scores: Both methods calculate the relative interestingness score and return as output the dominant measure(s), yielding the highest relative interestingness.

We measured the running times required to apply the Reference-Based and the Normalized interestingness comparison methods, w.r.t. each of these computation parts. For the Normalized Comparison, running times include the corresponding segment in the preprocess routine for each action.

Table 3 depicts the average time required by the Reference-Based and the Normalized comparison methods in order to select

Component	Time (Seconds)	
	Reference Based	Normalized
Action Execution	2.218	–
Calc. Interstingness	4.84	0.106
Calc. Relative Scores	0.04	0.031
Total	**7.2**	**0.138**

Table 3: Offline Running Times

Parameter	Value Range	Default Configuration	
		Ref. Based	Norm.
n-Context Size (n)	$[1, 11]$	4	2
kNN Size (k)	$[1, 40]$	1	1
Dist. Thres. (θ_δ)	$[0, 0.5]$	0.2	0.1
Int. Thres. (θ_I)	$[0, 1]$ (RB) $[-2.5, 2.5]$ (N)	0.92	0.7

Table 4: Model Hyper-Parameters

the dominant measure for a given action for each of the computation parts described above. First, observe that the Reference-Based requires an overall of 7.2 seconds compared to the Normalized which takes 0.138 seconds only. As for the computational parts, see that the Reference-Based requires a considerable amount of time for executing the alternative actions (Part 2) method (the average size of the reference actions set was 115) which is unneeded for the Normalized method. Consequently, the former requires computing all interestingness scores of the alternative actions, therefore its running time w.r.t. Part 1 is significantly longer than of the Normalized method. Part 3, in both methods is negligible.

4.2 Predictive Model Evaluation

We next describe our predictive kNN based model. We then explain how we evaluated its performance in comparison with several appropriate baselines.

Constructing the training set. **Extracting n-contexts.** We extracted *n*-contexts that belong to successful sessions from REACT-IDA repository using the DFS base method described in Section 3.2. We experimented with n-contexts of sizes 1 to 11 (we explain below how the default size was chosen). For each session state S_t, we stored the pair $\langle c_t, q_{t+1} \rangle$ comprising its context and the consecutive action.

Annotating n-contexts. We used the offline analysis results as described above, to label each pair $\langle c_t, q_{t+1} \rangle$ with its corresponding dominant interestingness measure $i^\star(q_{t+1})$. We then discarded all samples in which the maximal relative interestingness scores (obtained by $i^\star(q_{t+1})$) was smaller than the interestingness threshold θ_I (as described in Section 3.2).

In case that identical n-contexts obtained different labels[5] we unanimously labeled them by the most common label(s) associated with this n-context.

kNN Model implementation. As common for kNN based classification models, given a (non-labeled) n-context our model searches the training set for the top-k most similar labeled n-contexts, then selects the label by employing majority vote. To determine the similarity between n-contexts, we used the distance metric devised in [25] which was proven useful for IDA sessions. The metric is based on *tree edit-distance*, i.e. the minimum-cost sequence of *edit operations* (add, delete, and alter a node/edge) required to transform one n-context to the other. While a unit cost is given to delete/add operations, the cost of an *alter* operation (for a node/edge) is proportional to the similarity between the data displays and analysis actions. The latter is determined by two ground metrics for actions and displays: the first considers differences in the actions' syntax and the second measures the differences in the content of the compared displays.

As for the model's running times, we measured an average time of 6.04 milliseconds required to output a single prediction. We refrain from further discussing the computation costs and the scalability of the model and refer the reader to [25] for an in-depth performance evaluation of the nearest-neighbors search w.r.t. their distance metric.

Evaluation Methodology. We formed multiple test sets using the Leave-One-Out cross validation (LOOCV) method, i.e., in a single prediction task we take one sample (n-context) out of the training set to be used as a test set, then repeat the process for each and every sample.

We then used the following evaluation metrics: *(1) Accuracy*, which stands for the ratio between correctly predicted samples (true positives) and the number of all samples. Then, as is standard when evaluating multi-class classification models, we used the *(2) Macro-Averaged Precision* and *(3) Macro-Averaged Recall* measures, which takes the average of the precision (resp., recall) *w.r.t. each class* (i.e., each interestingness measure).[6] We also computed the *(4)Macro-Averaged F1* which is the harmonic mean of (2) and (3).

Last, since in some cases the kNN model does not output a prediction (recall from Section 3.2), we also measured the *(5) coverage rate*, namely the proportion of samples for which our model is able to produce a prediction.

Hyper-parameters tuning. Our model uses the following hyper-parameters: *(1) the size of n-contexts* used when constructing the training set to decide how many actions/displays are required to represent each session state S_t. *(2) The size of k*, i.e. the number of similar n-contexts used to perform a prediction. *(3) n-contexts distance threshold* θ_δ, representing the maximal distance allowed between each members of the kNN set and the given n-context. *(4) Interestingness Threshold* θ_I, i.e. the minimal relative interestingness score required for the sample to be considered as interesting and not to be discarded. Recall that relative interestingness is computed differently for each comparison method therefore this parameter has two sets of scales: For the Reference-Based method the threshold represents the minimal *percentile* rank of the actions in the reference set surpassed by the score of the given action (For example $\theta_I = 0.7$ means that we are interested in samples where the dominant measure ranks an action higher than at least 70% of the actions in the reference set). For the *Normalized* method, as the standardized scores largely falls between -2.5 and 2.5 standard deviations, θ_I represents the minimal number of standard deviations that the score should (positively) deviate from the mean.

To choose an optimal parameters configuration, we used a standard grid search consisting of more than $50K$ unique settings. Table 4 depicts the minimal, maximal and default value for each parameter, w.r.t. both interestingness comparison methods.

Since there is a tradeoff between the predictive performance and the coverage-rate (we explain this in the sequel), in order to choose an optimal configuration we calculated the skyline (also

[5] This can happen when users perform an identical subsequence of actions yet choose a different next action.

[6] In contrast, micro-averaged methods consider all true/false positives, regardless of the class.

	Reference-Based Comparison				Normalized Comparison			
	Accuracy	Macro-Precision	Macro-Recall	Macro-F1	Accuracy	Macro-Precision	Macro-Recall	Macro-F1
RANDOM	0.282	0.281	0.268	0.275	0.252	0.252	0.253	0.252
BestSM	0.397	0.397	0.250	0.306	0.329	0.329	0.250	0.284
I-SVM	0.632	0.636	0.482	0.549	0.655	0.674	0.617	0.643
I-kNN	**0.730**	**0.646**	**0.569**	**0.605**	**0.763**	**0.730**	**0.664**	**0.694**

Table 5: Interestingness Measure Prediction - Baseline Results

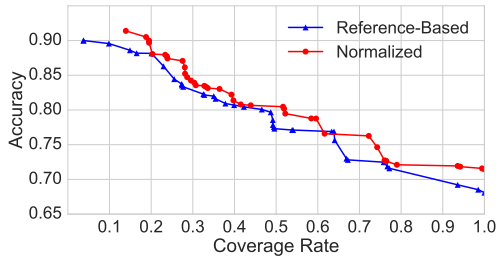


Figure 4: Configurations Skyline

called the Pareto frontier) for every comparison method. This resulted in a set of *dominant* configurations, w.r.t. the coverage and the accuracy/F1 score[7]. Figure 4 depicts the skyline plot for both comparison methods, between the coverage-rate (x-axis) and the accuracy (y-axis).

We chose default configurations from the skyline, as depicted in Table 4. Our default configurations yield accuracy scores of 0.730 and 0.763 (for the Reference-Based and Normalized methods, respectively), with coverage scores of 0.67 and 0.722 (resp.). In principle, any other configuration on the skyline can be chosen to ensure a different coverage-rate or predictive accuracy.

Baselines Comparison. We compared the performance of our predictive model, denoted *I-kNN*, with several other baselines.

(1) RANDOM, a naive baseline that selects a measure out of $\mathcal{I}$ uniformly at random. *(2) Best-SM*, this baseline chooses the *best single measure*, namely it always selects the one measure which is the most prevalent among the training set. This baseline corresponds to the common approach, taken in many analysis assistance tools (e.g. [10, 16, 31]), in which a single interestingness measure is chosen a-priori and used for all cases. Next, we experimented with two predictive models that are adequate to our setting in which the samples (namely, n-context) are compound objects rather than feature vectors: *(3) I-SVM*: a Support Vector Machine (SVM) model with a modified kernel [7] that can take an arbitrary distance matrix rather than using the Euclidean distance between vector-shaped samples. We used it with the dedicated distance metric for IDA sessions [25] described earlier in this section. We employed a standard grid search to tune the model's hyper parameters.

Table 5 lists the predictive evaluation scores for both the Reference-Based and Normalized comparison methods, averaged over all 16 measure combinations in $\mathcal{I}$.

First, we can see that the *Best-SM* baseline outperforms *RANDOM*, yet its accuracy is less than 40%. The latter reestablishes our first hypothesis that no single existing measure can adequately capture users' dynamic interestingness preferences, hence a-priori choosing a single interestingness measure in IDA may often lead to an erroneous outcome. Second, see that the *I-kNN* model outperforms the *I-SVM* demonstrating 14% higher accuracy and 10% higher F1 score. However, recall that in contrast to I-kNN (which uses the default configuration), the SVM base model obtains 100% coverage. Yet, as mentioned above, it is possible to choose a different configuration from the skyline (see Figure 4) to enforce full coverage. In such settings the improvement obtained by the kNN model over the SVM is less significant. Nevertheless, see that both predictive models *I-SVM* and *I-kNN* significantly outperforms *BEST-SM*, which corresponds to existing approaches to interestingness measures. This establishes our third hypothesis *that the right measure can be successfully predicted by examining the analysis n-context.*

In what comes next we examine the effect on the predictive performance (and coverage) of the model's hyper-parameters.

Hyper-parameters Effect. To study the effect of the model's hyper-parameters we repeated the predictive evaluation of the model while varying the values of each parameter and fixing the rest to their default values as depicted in Table 4.

In Figure 5 we present the Accuracy, Macro-F1 and the Coverage-Rate as a function of each of the system parameters (for both comparison method), where the other system parameters get their values from the default configuration (w.r.t. the comparison method used) in Table 4. We next examine the effect on the predictive performance and coverage when varying each of the model's parameters. Last, we present a summary of our findings.

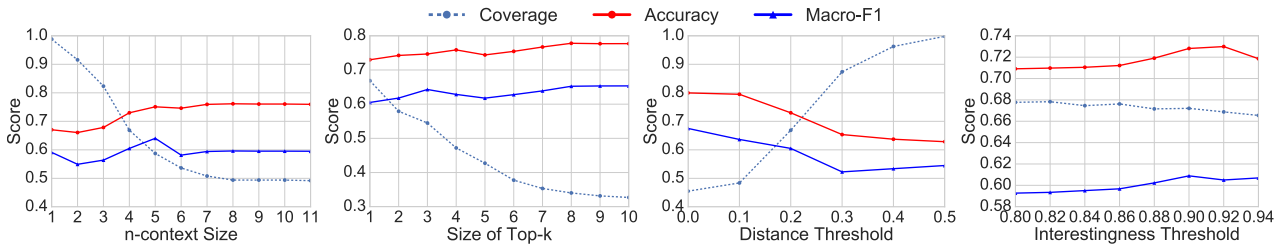
n-context Size. The n-context size, determined when preparing the training set, affects the amount of information the predictive models consider. Figure 5a1 and 5b1 depict the effect of n on the Accuracy, Macro-F1, and the Coverage-Rate of model for the Reference-Based and Normalized settings (resp.). As expected, increasing n (hence increasing the amount of information considered) positively affects the predictive performance. However, observe that the Coverage-Rate decreases. This is expected since when calculating the distance between larger, more compound, n-contexts the scores *increase* thus in more cases the k nearest-neighbors are not "similar enough" and the model does not output a prediction. When choosing n between 2-4, as in our default configurations, we obtained almost optimal predictive performance while retaining coverage of about 70% of the cases.

Size of k. The number of nearest neighbors considered by the model has a milder effect on performance, as can be seen in Figure 5a2 and 5b2 that demonstrate the effect of k on the Reference-Based and Normalized settings (resp.). In the Reference-Based method we can see a small, noticeable increase in performance, however it has a greater effect on the coverage of the model, since finding a larger set of nearest neighbors which are all similar enough to the given n-context is not always possible.

Distance Threshold θ_δ The distance threshold θ_δ is used by the model to enforce that the set of retrieved nearest neighbors are not too distant from the given n-context, hence avoiding erroneous predictions.

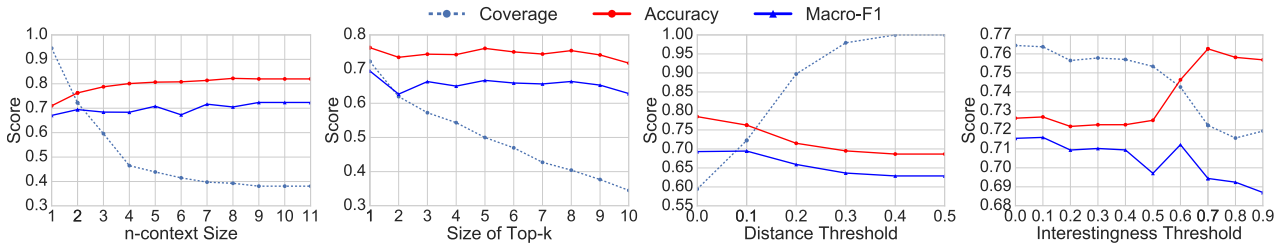
As expected, the lower (more tight) the distance threshold, the higher the predictive accuracy, as shown in Figures 5a3 and 5b3

[7] A configuration with x coverage and y accuracy is dominant if there is no other configuration with x' coverage and y' accuracy such that $x' \geq x \wedge y' > y$.



(a1) n-context Size **(a2) Size of Top-k** **(a3) Distance Thresh.** **(a4) Interestingness Thresh.**

(a) Reference-Based Comparison



(b1) n-context Size **(b2) Size of Top-k** **(b3) Distance Thresh.** **(b4) Interestingness Thresh.**

(b) Normalized Comparison

Figure 5: System Parameters Effect

displaying the effect of θ_δ when using the Reference-Based and the Normalized comparison methods. Naturally, the coverage decreases with θ_δ since there is not enough nearest neighbors with high similarity in many cases.

Interestingness Threshold θ_I. Recall that after we apply one of the offline comparison methods, we obtain the relative interestingness of each action. This allows us to filter out cases where the action executed by the user was not considered as interesting w.r.t. any of the measures in $\mathcal{I}$. For both comparison methods this indeed increases the predictive performance, as can be seen in Figures 5a4 and 5b4

Summary of Findings. We conclude this part by pointing out the trade-off between the predictive performance of the model and its coverage. When increasing the size of n-context and the kNN set size we increase the amount of information considered by the model, since a larger set of more comprehensive n-contexts are used as the basis for prediction. This naturally increases the model's predictive performance yet decreases the coverage since there are fewer cases where we can find, e.g., a large number of nearest neighbors highly similar to the given n-context.

A similar effect occurs when increasing the interestingness and similarity thresholds θ_I and θ_δ, which are used to ensure that the model uses only "high-quality" samples. Increasing theses thresholds improves the predictive performance yet decreases the Coverage-Rate, since there are fewer cases where such high-quality samples are relevant.

5 RELATED WORK

There is extensive literature that considers the interestingness of analysis actions on one hand, and interactive data analysis on the other.

Evaluating and comparing interestingness measures. As mentioned in the introduction to this work, interestingness measures are widely used in the field of data analysis, employed in tools e.g. for ranking and sorting association rules [17, 29], data patterns [4], generating useful visualizations [31], exploring OLAP data cubes [16] and many others.

Since dozens of measures were devised in the literature, each capturing different facets of the broad concept, several surveys and comparative studies have been performed to evaluate their usefulness [12, 14, 17, 18, 22, 29]: In these works the authors empirically evaluate many of the measures on real and synthetic datasets, and provide guidelines for choosing the right one for a given task and application domain. For example, [17] presents an empirical evaluation of more than 30 interestingness measures, applied for ranking buying patterns of customers. In [29], the authors examine 21 measures for association rules, concluding that neither one is consistently better than the others. [14] focuses on measures for data summary, empirically showing that the score distributions tend to be highly skewed.

While these notable studies contribute to the understanding of the usefulness of measures for different analysis tasks and scenarios, *they do not address the case of IDA - in which the interestingness criteria may dynamically change as the analysis session progresses.* To our knowledge, our work is the first to experimentally demonstrate this phenomenon and to provide a dynamic interestingness assessment (and selection of the appropriate measure), at each step of an ongoing analysis session.

Subjective facets of interestingness. For brevity, we considered in this work only *objective* facets of interestingness, such as Diversity, Peculiarity and Conciseness. However, several works suggest measures that capture *subjective* facets of interestingness [8, 20], i.e. that use prior information about the user and provide a more personalized interestingness assessment. For example, in [20] the authors devise measures for capturing interestingness facets such as *surprisingness* and *actionability* by considering user prior beliefs encoded as a set of classification rules (e.g. "has_job → loan_approved").

Incorporating such measures in our framework is possible yet requires the model to consider user information in addition to the n-contexts. This is an exciting direction for future research.

Learning Interestingness. Some works also suggest learning-based solutions for interestingness assessment. In [9] the authors present a system for guided data exploration based on active-learning. The system presents users with an initial set of tuples and asks then to annotate each tuple as interesting or not. Harvesting this feedback, the system can improve and personalize the tuples presented to the users. In [21], they present a visualization ranking system which is based on supervised binary classification of visualization into "interesting" or "non-interesting", based on students' annotations as ground truth.

In contrast to our work, these works tackle specific analysis tasks (i.e. filter/select queries and visualizations) hence can not be trivially generalized to the context of IDA which consists of sequences of actions of multiple types, and where (as we had shown in the experiments), interestingness, even for the same action on the same dataset, may vary through the process.

Second, both [9] and [21] require, and rely on users' feedback, which as explained in the introduction has limitations in our context. Different from these works, our solution provides a general-purpose system suitable for various types of analysis actions, and does not relay on particular user annotations. However, harnessing user feedback and learning-to-rank models in our system could be a promising direction for future work.

Mining IDA session-logs for action/query recommendation. Previous work [2, 11, 13, 25] suggests that mining IDA session logs can be used for predicting/recommending the next action in a session. These works utilize a collaborative-filtering approach, intuitively arguing that "if users are posing similar sequences of queries, they are likely interested in the same subpart of the dataset". However, in our previous work [25] we argue that in real life IDA scenarios analysts often examine different datasets from different purposes, hence recommending previous actions from the log to new users is generally impractical. To overcome this, the system described in [25] provides users with high-level "suggestions" that aggregate meaningful action-fragments mined from previous actions in the log. Combining our solution with [25] is an interesting direction for future work, where our solution can assist the system in [25] to better sort the output suggestions and further materialize them to concrete, executable actions.

6 CONCLUSION

This work examines interestingness measures in the context of interactive data analysis (IDA). We show that interestingness in IDA has unique characteristics: it dynamically changes even within a single session, and can not be holistically captured by just one measure. Using a real-life session log we demonstrated these characteristics and evaluated our interestingness predictive model, showing it can successfully select an appropriate interestingness measure for each step in an IDA session. Our model and framework may be employed in existing analysis recommender systems, allowing them to better fit the recommended next actions (e.g., visualizations, queries) to the current interestingness preferences of the user.

As for future work, we previously mentioned several ideas such as using our system to evaluate the effectiveness of analysis sessions. Also, incorporating user feedback and learning-to-rank models in our system is an exciting idea for future research.

ACKNOWLEDGEMENTS

This work has been partially funded by the Israel Innovation Authority, the Israel Science Foundation, Len Blavatnik and the Blavatnik Family foundation, and Intel®AI DevCloud.

REFERENCES

[1] REACT: Ida benchmark dataset. https://github.com/TAU-DB/REACT-IDA-Recommendation-benchmark.
[2] J. Aligon, E. Gallinucci, M. Golfarelli, P. Marcel, and S. Rizzi. A collaborative filtering approach for recommending olap sessions. *Decision Support Systems*, 69:20–30, 2015.
[3] J. Aligon, M. Golfarelli, P. Marcel, S. Rizzi, and E. Turricchia. Similarity measures for olap sessions. *KAIS*, 39, 2014.
[4] M. Boley, M. Mampaey, B. Kang, P. Tokmakov, and S. Wrobel. One click mining: Interactive local pattern discovery through implicit preference and performance learning. In *Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics*, pages 27–35. ACM, 2013.
[5] G. E. Box and D. R. Cox. An analysis of transformations. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 211–252, 1964.
[6] V. Chandola and V. Kumar. Summarization–compressing data into an informative representation. *Knowledge and Information Systems*, 12(3):355–378, 2007.
[7] Y. Chen, E. K. Garcia, M. R. Gupta, A. Rahimi, and L. Cazzanti. Similarity-based classification: Concepts and algorithms. *Journal of Machine Learning Research*, 10(Mar):747–776, 2009.
[8] T. De Bie. Subjective interestingness in exploratory data mining. In *Advances in Intelligent Data Analysis XII*, pages 19–31. Springer, 2013.
[9] K. Dimitriadou, O. Papaemmanouil, and Y. Diao. Aide: an active learning-based approach for interactive data exploration. *IEEE Transactions on Knowledge and Data Engineering*, 28(11):2842–2856, 2016.
[10] M. Drosou and E. Pitoura. Ymaldb: exploring relational databases via result-driven recommendations. *The VLDB Journal*, 22(6), 2013.
[11] M. Eirinaki, S. Abraham, N. Polyzotis, and N. Shaikh. Querie: Collaborative database exploration. *TKDE*, 2014.
[12] L. Geng and H. J. Hamilton. Interestingness measures for data mining: A survey. *ACM Computing Surveys (CSUR)*, 38(3):9, 2006.
[13] A. Giacometti, P. Marcel, and E. Negre. *Recommending multidimensional queries*. Springer Berlin Heidelberg, 2009.
[14] R. J. Hilderman and H. J. Hamilton. Evaluation of interestingness measures for ranking discovered knowledge. In *PAKDD*. Springer, 2001.
[15] R. J. Hilderman and H. J. Hamilton. *Knowledge discovery and measures of interest*, volume 638. Springer Science & Business Media, 2013.
[16] M. Joglekar, H. Garcia-Molina, and A. Parameswaran. Smart drill-down. *Target*, 6000:0, 2014.
[17] M. Kirchgessner, V. Leroy, S. Amer-Yahia, and S. Mishra. Testing interestingness measures in practice: A large-scale analysis of buying patterns. In *DSAA*, 2016.
[18] T.-D. B. Le and D. Lo. Beyond support and confidence: Exploring interestingness measures for rule-based specification mining. In *SANER*, 2015.
[19] S. Lin and D. E. Brown. An outlier-based data association method for linking criminal incidents. *Decision Support Systems*, 41(3):604–615, 2006.
[20] B. Liu, W. Hsu, L.-F. Mun, and H.-Y. Lee. Finding interesting patterns using user expectations. *IEEE Transactions on Knowledge and Data Engineering*, 11(6):817–832, 1999.
[21] Y. Luo, X. Qin, N. Tang, and G. Li. Deepeye: Towards automatic data visualization. ICDE, 2018.
[22] K. McGarry. A survey of interestingness measures for knowledge discovery. *The knowledge engineering review*, 20(1):39–61, 2005.
[23] T. Milo and A. Somech. React: Context-sensitive recommendations for data analysis. In *SIGMOD*, 2016.
[24] T. Milo and A. Somech. Deep reinforcement-learning framework for exploratory data analysis. In *Proceedings of the First International Workshop on Exploiting Artificial Intelligence Techniques for Data Management*, page 4. ACM, 2018.
[25] T. Milo and A. Somech. Next-step suggestions for modern interactive data analysis platforms. In *KDD*. ACM, 2018.
[26] J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.
[27] S. Sarawagi. User-adaptive exploration of multidimensional data. In *VLDB*, volume 2000, pages 307–316, 2000.
[28] S. Sarawagi, R. Agrawal, and N. Megiddo. Discovery-driven exploration of olap data cubes. In *ICDT*, 1998.
[29] P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the right interestingness measure for association patterns. In *SIGKDD*, 2002.
[30] T. C. Urdan. *Statistics in plain English*. Routledge, 2011.
[31] M. Vartak, S. Rahman, S. Madden, A. Parameswaran, and N. Polyzotis. Seedb: efficient data-driven visualization recommendations to support visual analytics. *VLDB*, 2015.
[32] Q. Yao, A. An, and X. Huang. Finding and analyzing database user sessions. In *International Conference on Database Systems for Advanced Applications*, pages 851–862. Springer, 2005.