A Recommendation System for Specifying and Achieving S.M.A.R.T. Goals

Romain Bardiau, Magali Seguran, Aline Senart and Ana Maria Tuta Osman

Business Intelligence Advanced Development, Sophia Antipolis, SAP Labs, Paris, France
{firstname, lastname}@sap.com

Keywords: Recommendation System, S.M.A.R.T. Goals, Forecast.

Abstract: Businesses and public organizations are typically goal-oriented trying to maximize their performance. Goals are today set arbitrarily and are given without hints on how to achieve them. There are many applications that allow setting up goals and sub-goals but the process is still manual. In this paper, we present a recommendation system that helps the user specify S.M.A.R.T. goals and monitor the progress towards these goals. Given a main goal on a metric, the system recommends specific sub-goals or indicators based on the forecast of historical data. These recommended indicators are the most probable to have a higher contribution in helping the user to reach his main goal. The user can additionally monitor its progress with a visualization over time. We show how this system can be used in a business scenario for sales.

1 INTRODUCTION

Businesses and public organizations are looking for new ways to maximize their precious resources while minimizing costs, especially in difficult economic contexts. Companies’ focus is typically on building sustainable competitive advantages around their strategic resources to increase their profit and revenue, regardless the size of their business. Public organizations top concerns are, on the other hand, to improve their service, maximize the use of their resources, and cut expenses.

One effective tool for making progress towards these desired results is to set up goals. Unfortunately, goals are today set arbitrarily and given without hints on how to achieve them. For example, a sales manager may give a goal to his team members like “increase customer satisfaction by 5 points” without any indication on where the effort should be taken, i.e., whether it should be on some specific products or on specific locations. It is up to the employee to find out how he will be able to achieve the goal by digging into and analyzing operational information.

Many applications today can be used for setting, tracking and ultimately achieving goals (Harris, 2013). The user can list his goals and associated tasks, set how often he wants to be reminded of them and can share them with his friends on social networks. However, the process to set up these goals is manual and there is no automated help provided to track their completeness. Some systems go a step further by dynamically suggesting goal optimizations but they are domain specific and mainly based on user monitoring (Digital, 2013).

In this paper, we present a recommendation system that helps the user specify a S.M.A.R.T. (Specific, Measurable, Assignable, Realistic, Time-related) goal (Meyer, 2003) and monitor the progress towards this goal. The system automatically recommends sub-goals or indicators to the user with minimal interaction. These recommended indicators are the most probable to have a higher contribution in helping the user to reach his main goal. To do so, the system forecasts how the company or organization is likely to perform for several activities based on past operational data and identifies where the effort should be taken to improve the forecasted situation.

The user can additionally monitor the progress of the indicators over time with a visualization widget. This monitoring helps the user discover at a glance the performance achieved. When progress suggests that the goal will unlikely be reached, adjustments of the goal can be made prior to the point in time when the goal should be accomplished.

To illustrate our recommendation system, we will develop a business scenario throughout this paper. We will consider Bob who is a sales representative in Spain. Given the low growth in sales last year for tablets, Bob decides to target a 10% increase in the first trimester of the coming year. We will show how
our solution will recommend indicators to help him reach his goal and will provide him with a widget that he can include on his favorite web page to monitor the progress.

The paper is organized as follows. Section 2 presents the related work. Section 3 presents the recommendation system, while Section 4 describes the architecture and implementation. Finally, Section 5 presents conclusions and future work.

2 RELATED WORK

Recommendation systems are usually using different filtering techniques to find the most interesting items for a user in a large space of possible options (Asanov, 2011). Content filtering techniques take into account the content of items to filter the ones that better match the user’s preferences or profile (Kalles et al., 2003). Collaborative filtering is another technique that identifies what items might be of interest for a particular user by looking at the interest of similar users (Sarwar et al., 2001; Herlocker et al., 2004). There are some hybrid works that employs user-based and item-based prediction to guess the rating that a user is going to provide for an item. For example (Papagelis and Plexousakis, 2004) leverages the logged history of ratings and content associated with users and items.

For goal setting, there are many applications that let the user manually set up goals, define sub-goals and monitor the progress from one sub-goal to another sub-goal (Gregory, 2010; GoalScape, 2013; Milestone Planner, 2013) but very few that provide recommendations to the user. Bridgeline Digital (Digital, 2013) proposes a Smart Recommendation Engine that dynamically suggests ways to optimize content for e-commerce. The recommendation engine helps to reach critical campaign goals through web analytics and user monitoring. They have a list of pre-established issues for each goal on which they are running functionality tests (e.g., download link/page not good, landing page does not drive to download).

In comparison, our solution does not need user profiling, logged history (which might lead to privacy issues), or predefined recommendations that usually cannot cover all the possible current and future goals for any domain. Our solution is based on past operational data to generate forecasts from which S.M.A.R.T. recommendations are made. Past operational data might originate from multiple sources and usually designs low-level data such as transactions and prices. To our knowledge, there is no prior work in the literature that uses prediction techniques in order to make recommendations based on forecasted operational data.

3 RECOMMENDATION SYSTEM

In the rest of the paper, we will assume to be in a business context where operational data is stored in data warehouses. In a data warehouse, data is organized hierarchically according to measures and dimensions. A measure is a numerical fact on operational data. Examples of measures for specific product sales data include quantity sold, revenue, and profit margin. Each measure can be categorized into one or more dimensions. Dimensions define categories of stored data. Examples of dimensions include time, location, and product. Each dimension can have one or more sub-dimensions. For example, the time dimension can include sub-dimensions of year, quarter, month, week, and so on.

3.1 Business Workflow

This section presents the business workflow detailing the different steps that are followed to generate goal recommendations.

3.1.1 Main Goal Setting

The user first defines a goal for a target measure by providing input to a GUI, as depicted in Figure 1. A quantitative value as point or percentage target needs to be entered for the goal. The user then selects the orientation of the defined goal (e.g., increase, decrease, maintain) from a drop down menu and specifies the time frame by selecting the dates in a calendar. The user finally selects a measure and possibly dimensions for the goal. The measures can be retrieved and ranked according to contextual data retrieved from a user profile (e.g., job, title, location) or with collaborative filtering techniques on historical data associated with the user (Liu et al., 2012). The user can alternatively select one measure from the set of all available measures in the data warehouse. The dimensions will act as filters and will restrict the scope of the selected measure.

Within the example business scenario discussed above, Bob can provide user input to define a goal for increasing sales growth by 10% from January 2014 to March 2014. More particularly, Bob can select the measure “sales growth” from the recommended measures, input “10” for the percentage target, select “increase” and set start and end dates. In Figure 1, the measures “margins”, “revenue” and “total sales”
were also available for selection in the "financial" category of the data warehouse. After providing his input, Bob can select the "Next" button to progress to recommended indicators.

### 3.1.2 Recommended Indicators

Once the user has defined his main goal, the system finds and recommends dimensions and subdimensions for the selected measure and associated filters. Figure 2 depicts an example GUI for displaying these recommendations and for enabling user selection of one or more recommendations. These recommendations are ranked based on the risk of the goal to be reached as it will be later explained in Section 3.2.

The user can select a dimension from the recommended dimensions selection menu and/or from the all dimensions menu. The dimensions listed in the latter menu include all available dimensions in the data warehouse. In the scenario, the recommended dimensions include dimension and subdimension "city, Barcelona" having a rank of 4.4 and other dimensions and sub-dimensions with a lower rank.

The combination of the selected measures, filters and dimensions define the indicator, which can be monitored over the selected time frame to assist the user in achieving the defined goal.

### 3.1.3 Goal Monitoring

Once an indicator has been fully specified by the user, the system generates a software widget that monitors the progress of the selected indicator towards reaching the goal (see Figure 3). A widget is a visual application or component comprising portable code intended to be used on different platforms. In addition, alerts are automatically set to inform the user when the trend has an undesired direction or reach a critical level, depending on user’s preferences. The alert can support the user to perform an action at a critical time in the timeframe to decrease the risk not to reach the goal. Note that the recommendation of actions is out of the scope of this paper.

In our scenario, if the user has selected "City, Barcelona" as indicator, then the system creates a widget to monitor the trend of the sales growth in Barcelona over time with alerts when the growth decreases. Bob adds the widget on his portal and enterprise blog to track the sales growth and share it with his colleagues.

### 3.2 Recommendation Model

Our recommendation model is based on a small set of concepts that are defined in this section.

First, we define risk as the probability of the recommended indicators to be above or below a certain threshold in the future. A forecast of operational data is performed to determine this risk. The threshold can be based on one or more known trends associated with
a measure. For example, the threshold can be the minimum value or a maximum value that is established for an indicator. In our case, the threshold is the value associated with the goal defined by the user (e.g., 10% in our scenario). The risk can be determined using a cumulative distribution function (CDF) based on the predicted trend and the threshold as follows:

\[
\text{risk} = CDF(\text{futureTrend}, \text{threshold})
\] (1)

We define the contribution of an indicator as the relative importance of the indicator in reaching the selected goal. The contribution can be specified manually by setting weights for indicators, or it can be automatically calculated. In our system, the contribution is calculated based on a predicted trend and one or more known trends:

\[
\text{contribution} = \frac{(\text{futureTrend} - \text{pastTrend})}{\sum(\text{futureTrend} - \text{pastTrend})}
\] (2)

Finally, the rank of an indicator can be determined based on a level of risk associated with the indicator and the contribution in reaching the defined goal:

\[
\text{rank} = \text{risk} \times \text{contribution}
\] (3)

4 IMPLEMENTATION

In this section, we will present the general architecture and then how it has been implemented on SAP HANA Appliance product (Farber et al., 2011).

4.1 Architecture

Figure 4 depicts the general architecture of the solution that is composed of three layers: a GUI layer, an engine layer, and a database layer. The GUI layer on the client-side provides interfaces from Section 3.1 that can be displayed on any computing devices (desktop, laptop, smartphones, etc.). The engine layer and the database layer are hosted on one or more servers. Communication between the client and the servers is over HTTP.

4.1.1 GUI Layer

The user interacts with a GUI of the GUI layer to generate input data that is provided to the engine layer. User input includes at least an operational goal, a timeframe, an orientation, a measure and a dimension.

4.1.2 Engine Layer

The engine layer contains the following components: an engine for storing user input, a forecast engine, a risk calculator, a contribution calculator, and a rank calculator. Data is automatically retrieved (e.g., at particular intervals) or selectively retrieved by one or more engines of the engine layer from one or more sources, processed and stored in the database layer.

Store User Input. This component receives data that has been input through the GUI and provides the data for storage in the user input data store. The user input data will be later consumed by several engines of the engine layer.

Forecast Engine. This component receives user input data and past operational data (e.g., stored in the past operational data store). Based on a prediction algorithm, it processes the user input data and the past operational data to generate a forecast that will be stored in the forecast data store.

Risk Calculator. This component determines a risk associated with the forecast based on the goal and orientation provided from the user input. The risk is stored in the dictionary data store.

Contribution Calculator. This component receives user input data and forecast data and processes the contribution associated with a respective indicator. The contribution is stored in the dictionary data store.
This component determines the rank of an indicator based on the level of risk associated with the indicator and the contribution in reaching the defined goal. The rank calculator stores the determined rank in the dictionary data store. One or more indicators and respective ranks can be displayed to the user in the GUI.

4.1.3 Database Layer

The database layer includes user input data store, past operational data store, forecast data store, and dictionary data store. These stores provide all the necessary tables.

4.2 Implementation in SAP HANA

We have developed the GUI layer with SAPUI5, a UI Development Toolkit for HTML5 (Network, 2013). This framework offers a series of libraries that frontend developers can use to build compelling HTML5-based applications. The recommendation service has been implemented as a native application running on the extended application services server (XS server) of SAP HANA Appliance for better performance. The native application calls two stored procedures running at the database level. The first one is a SQL procedure for calculating the forecasts based on the exponential smoothing algorithm from the Predictive Analytics Library of HANA. The second stored procedure calculates the risk and the contribution with R code executed in the SAP HANA database query execution plan. The input for these two stored procedures is found in a SAP analytical view and the output is stored for future retrieval in relational tables in the SAP HANA in-memory database system. Whenever the user is calling the UI for recommendations, SAPUI5 uses an OData service (Kirchhoff and Geihs, 2013) for querying the recommendation service.

5 CONCLUSION

The system presented in this paper recommends one or more indicators to be monitored for reaching a goal. More specifically, it enables users to define accurate and realistic goals, and supports user monitoring of the progress toward goals based on the one or more indicators that have been recommended. Indicators point to areas of improvement and can act as triggers for action for the user. The novelty of our approach is based on forecasting the past operational data for finding unfavorable trends in the future and make recommendations based on them.

In the future, we plan to introduce machine learning algorithms to provide better recommendations to the user based on the usage of our recommendation system. We would also like also to leverage the ERP systems to automatically propose templates for differ-
ent business lines.

REFERENCES


