High Level Design (HLD)

Wine Reviews Analysis

Revision Number: 1.0
Last date of revision: 04/04/2022

Koshal Kumar

Document Version Control

Date Issued	Version	Description	Author
04th Apr 2022	1.0	First Version of Complete HLD	Koshal Kumar

Contents

Abstract

The purpose of this paper is to provide a tutorial of data analysis methods for answering questions that arise in analyzing data from wine-tasting events: (i) measuring agreement of two judges and its extension to m judges; (ii) making comparisons of judges across years; (iii) comparing two wines; (iv) designing tasting procedures to reduce burden of multiple tastings; (v) ranking of judges; and (vi) assessing causes of disagreement. In each case we describe one or more analyses and make recommendations on the conditions of use for each.

1 Introduction

1.1 Why this High-Level Design Document?

The purpose of this High-Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding, and can be used as a reference manual for how the modules interact at a high level.

The HLD will:

- Present all of the design aspects and define them in detail
- Describe the user interface being implemented
- Describe the hardware and software interfaces
- Describe the performance requirements
- Include design features and the architecture of the project
- List and describe the non-functional attributes like:
  - Security
  - Reliability
  - Maintainability
  - Portability
  - Reusability
  - Application compatibility
  - Resource utilization
  - Serviceability

1.2 Scope

The HLD documentation presents the structure of the system, such as the database architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly-technical terms which should be understandable to the administrators of the system.

2 General Description

2.1 Product Perspective & Problem Statement

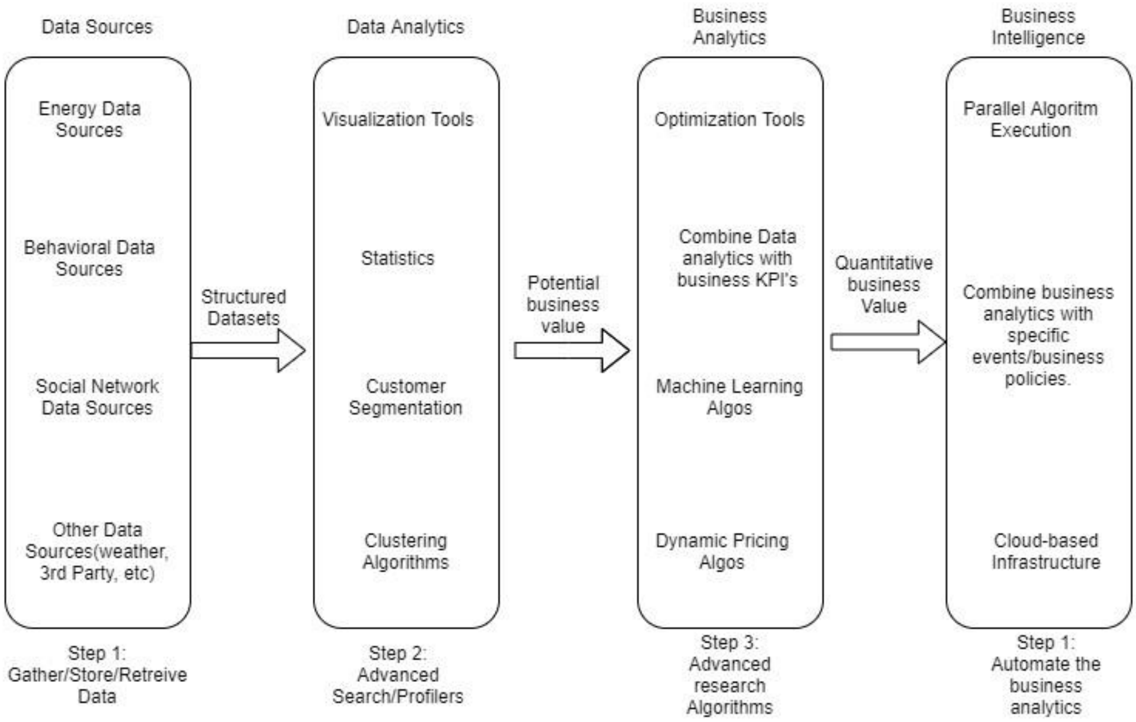
While the wine industry generally has been very proactive about dealing with climate change, from capturing fermentation carbon to trialing new varieties, this is ultimately a political problem. To find an effective solution for this problem you were asked to help in ETL process.

2.2 Tools used

Business Intelligence tools and libraries works such as Numpy, Pandas, Seaborn, Matplotlib, Excel, R, Python are used to build the whole framework.

3 Design Details

3.1 Functional Architecture



Data Sources
Energy Data Sources
Behavioral Data Sources
Social Network Data Sources
Other Data Sources(weather, 3rd Party, etc)
Step 1: Gather/Store/Retreive Data

Structured Datasets

Data Analytics
Visualization Tools
Statistics
Customer Segmentation
Clustering Algorithms
Step 2: Advanced Search/Profilers

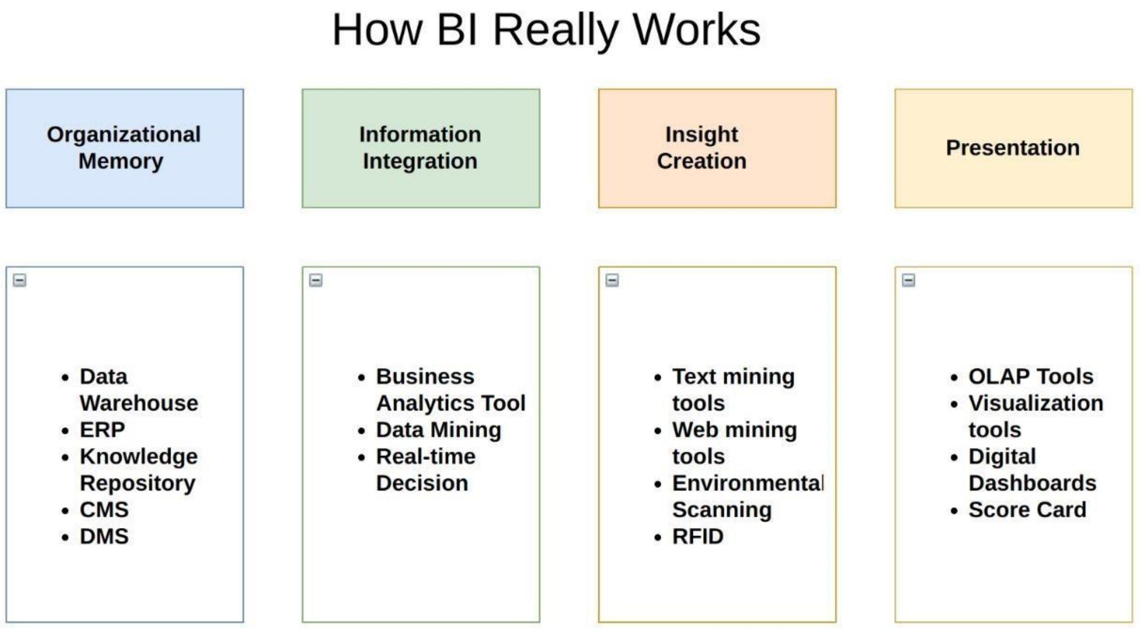
Potential business value

Business Analytics
Optimization Tools
Combine Data analytics with business KPI's
Machine Learning Algos
Dynamic Pricing Algos
Step 3: Advanced research Algorithms

Quantitative business Value

Business Intelligence
Parallel Algoritm Execution
Combine business analytics with specific events/business policies.
Cloud-based Infrastructure
Step 1: Automate the business analytics

Figure 1: Functional Architecture of Business Intelligence



How BI Really Works

Organizational Memory

- Data Warehouse
- ERP
- Knowledge Repository
- CMS
- DMS

Information Integration

- Business Analytics Tool
- Data Mining
- Real-time Decision

Insight Creation

- Text mining tools
- Web mining tools
- Environmental Scanning
- RFID

Presentation

- OLAP Tools
- Visualization tools
- Digital Dashboards
- Score Card

3.2 Optimization

Your data strategy drives performance

- Minimize the number of fields
- Minimize the number of records
- Optimize extracts to speed up future queries by materializing calculations, removing columns and the use of accelerated views

Reduce the marks (data points) in your view

- Practice guided analytics. There's no need to fit everything you plan to show in a single view. Compile related views and connect them with action filters to travel from overview to highly-granular views at the speed of thought.
- Remove unneeded dimensions from the detail shelf.
- Explore. Try displaying your data in different types of views. **Limit your filters by number and type**

- Reduce the number of filters in use. Excessive filters on a view will create a more complex query, which takes longer to return results. Double-check your filters and remove any that aren't necessary.

- Use an include filter. Exclude filters load the entire domain of a dimension, while include filters do not. An include filter runs much faster than an exclude filter, especially for dimensions with many members.

- Use a continuous date filter. Continuous date filters (relative and range-of-date filters) can take advantage of the indexing properties in your database and are faster than discrete date filters.

- Use Boolean or numeric filters. Computers process integers and Booleans (t/f) much faster than strings.
- Use parameters and action filters. These reduce the query load (and work across data sources).

Optimize and materialize your calculations

- Perform calculations in the database
- Reduce the number of nested calculations.
- Reduce the granularity of LOD or table calculations in the view. The more granular the calculation, the longer it takes.
    - LODs - Look at the number of unique dimension members in the calculation.
    - Table Calculations - the more marks in the view, the longer it will take to calculate.
- Where possible, use MIN or MAX instead of AVG. AVG requires more processing than MIN or MAX. Often rows will be duplicated and display the same result with MIN, MAX, or AVG.
- Make groups with calculations. Like include filters, calculated groups load only named members of the domain, whereas Tableau's group function loads the entire domain.
- Use Booleans or numeric calculations instead of string calculations. Computers can process integers and Booleans (t/f) much faster than strings. Boolean>Int>Float>Date>DateTime>String

4 KPIs

Dashboards will be implemented to display and indicate certain KPIs and relevant indicators for the disease.

matplotlib seaborn

As and when, the system starts to capture the historical/periodic data for a user, the dashboards will be included to display charts over time with progress on various indicators or factors.

4.1 KPIs (Key Performance Indicators)

Key indicators displaying a summary of the Housing Price and its relationship with different metrics

1. Impact of Points on Wine Quality
2. Impact of Price on House Wine Quality
3. Influence of Province parameter on Wine Quality
4. Influence of Variety parameter on Wine Quality
5. Influence of Winery parameter on Wine Quality
6. Influence of Country parameter on Wine Quality

5 Deployment

Prioritizing data and analytics couldn't come at a better time. Your company, no matter what size, is already collecting data and most likely analyzing just a portion of it to solve business problems, gain competitive advantages, and drive enterprise transformation. With the explosive growth of enterprise data, database technologies, and the high demand for analytical skills, today's most effective IT organizations have shifted their focus to enabling self-service by deploying and operating Tableau at scale, as well as organizing, orchestrating, and unifying disparate sources of data for business users and experts alike to author and consume content.

Tableau prioritizes choice in flexibility to fit, rather than dictate, your enterprise architecture. Tableau Server and Tableau Online leverage your existing technology investments and integrate into your IT infrastructure to provide a self-service, modern analytics platform for your users. With on-premises, cloud, and hosted options, there is a version of Tableau to match your requirements. Below is a comparison of the three types:

TYPE PROS CONS

It may sometimes seem easier to go through a set of data points and build insights from it but usually this process may not yield good results. There could be a lot of things left undiscovered as a result of this process. Additionally, most of the data sets used in real life are too big to do any analysis manually. This is essentially where data visualization steps in.

Data visualization is an easier way of presenting the data, however complex it is, to analyze trends and relationships amongst variables with the help of pictorial representation.

The following are the advantages of Data Visualization

- Easier representation of compels data
- Highlights good and bad performing areas
- Explores relationship between data points
- Identifies data patterns even for larger data points

While building visualization, it is always a good practice to keep some below mentioned points in mind

- Ensure appropriate usage of shapes, colors, and size while building visualization
- Plots/graphs using a co-ordinate system are more pronounced
- Knowledge of suitable plot with respect to the data types brings more clarity to the information
- Usage of labels, titles, legends and pointers passes seamless information the wider audience

Matplotlib

It is an amazing visualization library in Python for 2D plots of arrays, It is a multi-platform data visualization library built on *NumPy* arrays and designed to work with the broader *SciPy* stack. It was introduced by John Hunter in the year 2002. Let's try to understand some of the benefits and features of *matplotlib*

- It's fast, efficient as it is based on *numpy* and also easier to build
- Has undergone a lot of improvements from the open source community since inception and hence a better library having advanced features as well
- Well maintained visualization output with high quality graphics draws a lot of users to it
- Basic as well as advanced charts could be very easily built
- From the users/developers point of view, since it has a large community support, resolving issues and debugging becomes much easier

Seaborn

Conceptualized and built originally at the Stanford University, this library sits on top of *matplotlib*. In a sense, it has some flavors of *matplotlib* while from the visualization point, its is much better than *matplotlib* and has added features as well. Below are its advantages

- Built-in themes aid better visualization
- Statistical functions aiding better data insights
- Better aesthetics and built-in plots
- Helpful documentation with effective examples