

ORIGINAL MOTION PICTURE SCORES

All Music Composed And Conducted By MIKLÓS RÓZSA

# QUO VADIS



Met. Madame Bovary | Ivanhoe | Plymouth Adventure

TSUNAMI

# Visualization Tools market is shrinking



Feb 6th, 2016 Tableau lost 50% of its market evaluation.  
BI market does not need another Tableau, QlikView.  
-source Fortune Magazine

BellaDati is NOT only visualization tool

BellaDati IS Analytics Framework & API  
that includes also Agile BI. We give our  
partners power to build own solution  
quickly

2016 roadmap to provide even more powerful:

- I. FRAMEWORK API - CLIENT API, REST API, SDK
- II. MACHINE LEARNING (IoT) studio with ready industry packages
- III. MACHINE LEARNING(IoT) domain specific language
- IV. BIG DATA for MACHINE LEARNING and STATISTICS using Spark (significantly faster than R Language based solutions)

# Why BellaDati Analytics Framework?

Save money. Deliver fast. Increase profit.

BellaDati framework solved for you the rocket science

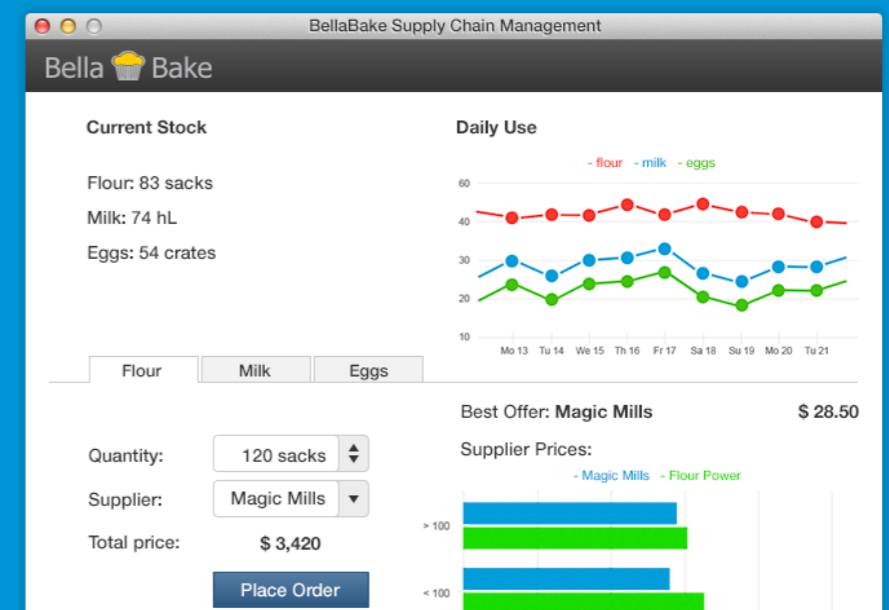
You have to write only few lines. Stay solution focused.

```
package com.belladati.bi.web.chart;
import java.util.HashSet;
/**
 * @author Denis Stepanov
 * @author Lubomir Misko
 */
public class GanttChartRenderer2 extends ChartElementRenderer {
    private HorizontalBarChart membersChart;
    private MultiBar multiBar;
    private ReadablePartial dateFrom, dateTo; //starting/ending date of chart - processed dynamically
    private Set<SelectedClassifiers> processed;
    @Override
    public void createChart(ChartDataContext context, AxesRenderer axesRenderer) {
        DateTimeAxisChartElement dateTimeElement = context.getChartView().getDateTimeAxisChartElementOrCreate();
        dateTimeElement.setDateAggregation(DateAggregation.DayMonthYear);
        dateTimeElement.setDateShowAllValues(true);
        AttributeAxisChartElement element = context.getChartView().getAttributeAxisChartElement();
        if (element != null && (element.getLimit() == null || element.getLimit() > 200)) {
            element.setLimit(200);
        }
        context = (ChartDataContext) context.withDateInterval(context.getChartView().getDateIntervalDefinition());
        DateTimeAxisRenderer dateTimeAxisRenderer = new DateTimeAxisRenderer(context, dateTimeElement);
        context = context.withAxesDefinition(new AxesDefinition(dateTimeAxisRenderer));
        Chart chart = context.getChartView().getChart();
        axesRenderer.createXVAxes(chart, context);
        ChartRenderingPipelineBuilder builder = renderingPipelineBuilder();
        builder.build(context, builder);
        builder.build().execute(context);
        if (context.getChartView().getDateIntervalDefinition() != null) //reflect the date interval settings if available, take the chart range otherwise
            if (context.getChartView().getDateIntervalDefinition().getFrom() != null) {
                dateFrom = context.getChartView().getDateIntervalDefinition().getFrom();
            }
            if (context.getChartView().getDateIntervalDefinition().getTo() != null) {
                dateTo = context.getChartView().getDateIntervalDefinition().getTo();
            }
        }
        dateTimeAxisRenderer = new DateTimeAxisRenderer(dateTimeElement) {
            @Override
            public ChartPipelineStage getStage() {
                return new DateTimeFromToValuesStage(new LocalDate(dateFrom), new LocalDate(dateTo));
            }
        };
        axesRenderer.addXAxisLabels(context, dateTimeAxisRenderer);
        axesRenderer.addYAxisLabels(context, context.getAxes().getY());
        if (context.getChartView().getXAxisHighlightToday()) {

```

Tons of code. Math. Analytics. ML. Big Data. Data cleaning.. algorithms

```
var api_call = "reports/views/152-Pzbf19zbzm/image?width=600&height=400"
```



# Machine Learning studio combined with Framework API + Agile BI + ETL + DWH

The screenshot displays the BellaDati ML Studio interface. On the left, a sidebar lists various functions under 'Functions (16)', including Customer Segmentation, Stock Segmentation, HR Employee segmentation, Credit scoring, Credit prediction, Market basket prediction, Predictive maintenance, Boxplot, Standardized boxplot, Normalization, Clustering, Standard deviations, Summary, Correlation, Linear regression, and Customer churn prediction. The main area shows a code editor with a Scala script for clustering. The script defines a 'segment' function that reads a CSV file, normalizes the data, performs clustering, and generates summary statistics and boxplots for each cluster. The output shows a boxplot for 'cluster 3 - normalized - summary relative' with categories: Intern, Calls, Text, Data, and Age. The y-axis represents percentages from 0% to 100%. A red arrow points from the 'BUILD FUNCTION' button to the boxplot, and another red arrow points from the 'Data' category to the text 'See ML results in realtime'. A third red arrow points from the code editor to the text 'Build your customized solution'.

ML Studio - Segmentation and clustering

```
1 def segment(file, numberOfClusters, numberOfIterations) {
2   def s = summary( { readCSVFile(file, ',', it) } )
3
4   //boxPlotChart(s, [ "color" : "red", "start" : 1, "labels" : 0, "exclude" : [ "Text" ] ])
5
6   //retrieve the first row of the file == header
7   def cols
8   readCSVFile(file, ',') {
9     cols = columns
10  }
11
12  //normalize
13  def data = normalize( { readCSVFile(file, ',', it) } )
14  //print out the normalized data
15  //table(data, cols)
16
17  //clustering
18  Dataset[] clusters = cluster(data, numberOfClusters, numberOfIterations)
19
20  //print out all clusters
21  for (int i = 0; i < clusters.length; i++) {
22    clusterId = 'cluster '+String.valueOf(i+1) + ' - normalized '
23    table(clusterId, clusters[i], cols)
24
25    sum = summary2( { readTable(clusterId, it) } )
26    boxPlotChart(clusterId + " - summary relative", sum, [ "start" : 1, "labels" : 0, "scale" : "relative", "outCo
27    boxPlotChart(clusterId + " - summary normal", sum, [ "start" : 1, "labels" : 0, "outCol" : 7 ] )
28  }
29
30  //measure the quality of the clustering
31  ClusterEvaluation sse = new SumOfSquaredErrors();
32  table("score", [ sse.score(clusters) ], "score" )
33
34  //print out denormalized
35  denormalize(s, cols, clusters)
36
37  return null
38 }
39
40 def denormalize(s, cols, clusters) {
41   /*
42   denormalize
43   CR = MAX-MIN
44   CM = (MIN+MAX) / 2
45   NormalizeMidrange:
46   y = (x - CM) / CR * R + M
47   x = (y - M) * CR / R + CM
48   */
49   java.text.DecimalFormat df = new java.text.DecimalFormat("##0.000")
50   list + new AppendList()
51 }
```

cluster 3 - normalized - summary relative

Category	Min	Q1	Median	Q3	Max
Intern	40%	42%	58%	82%	100%
Calls	8%	50%	68%	86%	100%
Text	0%	0%	6%	10%	100%
Data	28%	40%	45%	48%	60%
Age	28%	48%	55%	63%	82%

Processed in 9804ms.  
Result is null.

Build your customized solution

See ML results in realtime

No coding ready to use functions

Build your customized solution

See ML results in realtime

# Complete Advanced Analytics Tool Out of the Box and Platform = Key Differentiation

**1**  Rich **REST API** and **SDK**


---


**2**  Rich **Client API**

---

**3**  Ready made for **embedding**

---

**4**  **ETL** tool (Data Cleaning)

**5**  **Machine learning** and **Predictions**. Simple and complex **Algorithms**.

---

**6**  **IoT Framework**

---

**7**  **Data mining** - light and heavy weight. **Packages** and **studio**.

---

**8**  **Big Data** (Hadoop & MongoDB)