**2015/7/13　　林涛**

**Zeppelin**

Big Data Visualization using Apache Spark and Zeppelin

Prajod Vettiyattil, Software Architect, Wipro

https://zeppelin.incubator.apache.org/

Apache Zeppelin

A web-based notebook that enables interactive data analytics.

You can make beautiful data-driven, interactive and collaborative documents with SQL, Scala and more.

Demo：

https://www.youtube.com/watch?v=_PQbVH_aO5E&feature=youtu.be

这个主要是能在网页上写 spark 或 sql 脚本并将执行结果返回到前端。

对返回的数据能做简单的可视化，或者说更倾向于统计，仅限于内置好的几个图表类型。

能交互，修改代码重新计算。

这是个开源项目 ,能看到现在开发中的原码 ,但尚未 release。我在尝试将其 build 出来试验一下。

**Hossein Falaki 的项目**

http://strataconf.com/stratany2014/public/schedule/detail/36753

Interactive Visual Data Exploration with Spark

Hossein Falaki

（有 Summary）

https://spark-summit.org/east-2015/talk/visualizing-big-data-in-the-browser-using-spark

Visualizing big data in the browser using Spark

Hossein Falaki

（有 slides 和 video。）

这两个演讲中的展示的是同一个项目，是 Databricks 在做的。

从 demo 上看，这个项目的框架应该就是基于 Zeppelin 的，但在可视化方面更进了一步。

他的思路是将数据操作与渲染分离开来。先将数据用 spark 算出来，然后使用 ggplot 或 d3 可视化。

对于大数据的可视化，遇到的问题主要是：处理大数据要花较长时间，数据点可能比像素还多。因此他把数据的操作概况了三种思路：

a) “Summarize and visualize”

使用 Spark 进行 Aggregation 和 Pivoting

b) “Sample and visualize”

使用 Spark 原生提供的采样和分层采样。

c) “Model and visualize”

使用 Spark 的 MLLib 提供的一些用于机器学习的算法，如聚类、降维、假设检验。

后面是一个演示：

所有可见的操作都是在网页上完成的，编辑好脚本后让后端去计算，计算结果再

返回给前端进行了显示。

用 ggplot 的话，算和画都用 python 作为脚本。

用 d3 的话，将计算结果存成文件。然后 displayHTML。在线改 d3 的代码能看到新的结果。

他并没有针对 ggplot 或 d3 做出特别的优化，仅仅是用这些可视化库调用了计算的结果进行绘制。

演示中有个是关于大图的，利用 spark 能快就构建出 graph。但到要呈现的时候，也没什么优化，只是用 spark 将前面算出的 graph 采样了一下（根据一个字段筛选出了要的 edge），最后要呈现的数据还是挺小的。他将这些数据保存进了一个 js 文件，然后贴了一段 d3 的代码将其可视化出来。

我估计这个项目最后会集成到 Zeppelin 中去。

我觉得这个项目对于打通前后端这件事已经做得非常完善了，整个已经形成了 pipeline。但是对可视化的支持仅停留在数据层面，并没有将可视化的范式与数据紧密结合起来。

在这个完成度这么高的项目面前，我觉得我们照做整个系统不大现实也没什么意义，可以针对一些问题做突破：

1. 在这个项目中，交互主要是看到结果后修改代码（包括筛选的条件）重新计算。因此每次看到结果都要是计算完成之后。跟之前谈的想法那样，我们可以针对那些计算没那么快的更大的任务，做**分层次**的计算。

2. 将**交叉检索**的思路进去。在这个项目中，信息流只是从后端到前端的，如果要交互的话，要不是将需要的数据全放在前端用前端实现，要不就是手工改后端的代码重新计算。我们可以做到将前端的 brash 结果再返回到后端，形成 loop。

3. 或许可以针对要可视化的数据的情况提供对应的可视化形式，把底层透明掉？这点目前只是想想而已。