Budapest University of Technology and Economics
Department of Measurement and Information Systems
Fault Tolerant Systems Research Group

Critical Architectures Laboratory (Spring 2016/17)

Monitoring Data Analysis

Imre Kocsis, based on the previous iteration of the lab by Ágnes Salánki
V1.0, April 5, 2017.

1 Introduction

Data-driven approaches have always been present in the design, evaluation and operation of critical systems; however, the relatively recent emergence of "data science" is rapidly making them much more popular, easy to perform and accessible for the average practitioner. The main goal of this lab is to give a rather bird's-eye overview of applying modern, general pupose data analysis approaches and tools to the data analysis of critical infrastructures.

More specifically, our intention is to give an example-based introduction to the following topics:

1. The common contemporary idioms of data analysis in general purpose scripting and programming languages; specifically, the "data frame " abstraction and the most common operations over data frames.

2. Notebook-based data analysis. (Interactive) notebooks constitute a relatively new, but very important addition to the list of generic data analysis styles (classically mostly REPL shells/IDEs, explicit workflows, fully interactive data exploration tools and BI-style dashboards).

3. Plotting using rich library support. In the not so distant days of the past, general purpose data visualization in the practice meant picking a potentially closed, dedicated tool from a rather small set - most probably Excel, Matlab or gnuplot. One could argue that visually, the results were a bit lacking. In contrast, we have now a very wide array of options; conceptually one of the main advances is that visualization is getting very well "integrated into" analytic scripting and programming.

4. Integrated (and relatively painless) access to Big Data capabilities. "Big Data" tools have originally emerged to enable relatively cheap (at least compared ot data warehouses) distributed processing of massive amounts of "data at rest" - i.e. data that will not be updated or deleted from. However, this broke the convenient status quo of "everything that stores data speaks SQL"; for a while, analysis of Big Data meant that the "data scientist" had to essentially perform at least some algorithm development as well as code development for the Big Data platform at hand - to achieve functionality that has been rather trivial in classis SQL-based as well as statistical scripting settings. A welcome development is that this is changing, too; in addition to Big Data tools gaining true SQL support, in a number of important cases access to Big Data processing is now very well integrated into "workstation-based" statistical scripting and programming.

2 Python or R?

The development of data science is now at a point where all our goals could be demonstrated on wide range of platforms and using a wide range of contemporary scripting and programming languages (e.g. manifestions of Data Frames are even present in C# and Java now). As our data analysis examples are geared towards data exploration, we focus on scripting languages.

Instead of specifying a language, this year the lab has two "tracks" in the sense that you are allowed to choose between Python and R, based on your existing experience. That being said, if you are not familiar with R, you are strongly suggested to do the lab preparation and the lab itself in Python.

3 Preparation for the lab

As a preparation for the lab, you are expected to create an account for the free service https://datascientistworkbench.com/ (BDU Labs); we plan to use this platform. We want to emphasize that - with some minor exceptions - the tools that are bundled together in this Software as a Service solution are all free and open source; if you are so inclined, you are welcome to roll and bring along your own environment (e.g. using Docker Hub images this should not be particularly hard).

Additionally, you should familiarize yourself with the following material. We do not expect full proficiency; however, to finish the lab assignment in a timely manner you *will* have to know where to look and what to search for.

3.1 Jupyter/IPython

On BDU Labs, perform the "The Basics" tutorials. The examples are for Python, but with some existing experience in R can be easily translated for the R kernel. Note that while R is an option, RStudio, although it is present in the SaaS bundle, is not; you will be expected to work in Jupyter (with the R kernel).

3.2 Pandas

From the official pandas documentation: http://pandas.pydata.org/pandas-docs/stable/tutorials.html, review the "10 minute tutorial" then proceed to lessons 1 through 6 from "Lessons for New pandas Users" (you can skip the Excel-specific parts). Reading the Cookbook may be advantegous, but is not essential. On the R side: as data frames are a language-level concept in R, if you know R, you should know at least one way to perform what the referred pandas material talks about. That being said, you can benefit from looking at the package dplyr and/or data.tables.

3.3 Pandas visualization (matplotlib with training wheels)

With matplotlib under the hood, Pandas has decent built-in visualization support - and this should be enough for the lab assignments. Note that while you are welcome to use matplotlib more directly (or use the Python visualization library of your choice), struggling with the peculiarities of a visualization library can become a time sink very fast in any language.

You should review the respective part of the pandas documentation: http://pandas.pydata.org/pandas-docs/version/0.19.0/visualization.html Note: you are *not* expected to read and know this in detail; however, you should be aware of your options (what can be done and what can't). For R: ggplot2 is always a safe bet; base graphics will be frowned upon.

Note: due to your repeated exposure in the courses of the research group, we will expect you to know the basics (philosphy, approach, typical plot types) of Exploratory Data Analysis (EDA). You can find some Hungarian reference material from our group here:

https://inf.mit.bme.hu/sites/default/files/materials/category/kateg%C3%B3ria/oktat%C3%A1s/bsc-t%C3%A1rgyak/rendszermodellez%C3%A9s/16/08-vizualis-elemzes.pdf

and here:

http://www.interkonyv.hu/konyvek/antal_peter_intelligens_adatelemzes

Note that both sources approach EDA with a heavy emphasis on its "visual analysis" aspect. (This is due to the fact that your curriculum does not include statistics - so discussing the role and place of EDA in the statistical workflow in detail would lack the necessary underpinnings.)

3.4 Notebook-integrated Big Data: Spark

The lab will include a "Big Data" assignment - infrastructure monitoring data is a prime example of (valuable) "at rest" data for which bulk storage and distributed processing on "clusters of grey boxes" makes the most sense.

We will use one of the currently most successful Big Data platforms, Apache Spark. However, neither the scope nor the time constraints of the lab make an in-depth treatment of the subject possible. What we aim at is demonstrating that a) when offered in a Platform as a Service style, Big Data services can be integrated into an application in a rather "painless" way (i.e. no need to acquire computing and

storage resources, setting up the platform and configuring it); b) Big Data services have good language integrations that enable "ingesting" them in an analytic script easily. Consequently, while you are advised to gain familiarity with Spark *on the level of basic concepts*: http://spark.apache.org/talks/overview.pdf, we do not expect any deep proficiency. Spark has good (data frame) APIs that we will program against; review the "Spark SQL, DataFrames and Datasets Guide": https://spark.apache.org/docs/latest/sql-programming-guide.html#getting-started up to the Data Sources part. The lab assignment document will help you with getting around the specifics.

4 Self-check questions

If you are well-prepared for the lab, you should be able to answer the following questions:

1. From the point of view of data analysis workflow, how does the "Notebook" abstraction of Jupyter/IPython compare to classic tools running a REPL shell? (MATLAB, standard R, standard interactive command line Python, ...). What are the main benefits of Notebooks?

2. What is a data frame? (Either a) in general, b) specifically in pandas or c) specifically in R).

3. What are the main ways to filter a data frame in pandas/in R for a set of rows/set of columns?

4. How would you perform the "group by" operation on a data frame in pandas/R?

5. Specify a way for appending a new column to a data frame in pandas/R!

6. How are missing values represented in pandas/R?

7. What are the fundamental visualizations for a single observed variable? (Recall that data can be categorical as well as numeric).

8. What are the fundamental visualizations for two observed variables? (Recall that data can be categorical as well as numeric).

9. What is an RDD in Spark?

10. What are the most important Spark RDD Transformations?

11. What are the most important Spark RDD Actions?

12. Describe the difference between exploratory and confirmatory data analysis!

5 Apache Virtual Computing Lab data

We will work with the reservation and platform metric data of our production Apache Virtual Computing Lab cluster. For an overview of the data, please refer to last year's syllabus: https://inf.mit.bme.hu/sites/default/files/materials/category/kateg%C3%B3ria/oktat%C3%A1s/msc-t%C3%A1rgyak/kritikus-architektu%CC%81ra%CC%81k-laborato%CC%81rium/16/ca-monitoring-data-analysis.pdf

6 Lab assignments

The description of the lab assignments is geared towards Python users. If you want to use R (recommended only to advanced R users) – ask your lab instructor if you get stuck.

6.1 What you should hand in

Please hand in (i.e. upload to GitHub) the notebook definition file as well as a full html/pdf rendering of your final solution.

Take care to explain what you are doing (use the built-in Markdown facilities!) and explain the results, where appropriate. The mapping between the notebook cells and the assignments below should be evident. Please don't change the input file names.

6.2 Assignments

6.2.1 Data preparation

1. Get the file reservations.csv and upload it in the "My Data" sub-service.
2. Create a new Notebook with the kernel you chose (Python 2, Python 3 or R).
3. Read the csv into a data frame. The path to your data should begin with /resources/data. Note that Jupyter supports tab completion for many things - file paths as well as functions and their arguments.
4. Print the first few lines of the data frame.
5. Convert the columns representing time to pandas's datetime type! Hint: pandas has a very serviceable to_datetime function.
6. Ensure that the able_to_serve column is boolean! Hint: this is easy using the map function that is available for columns.
7. Print the basic descriptive statistics for each column! Hint: pandas data frames have a good "describe" function, however, you should split the analysis for column type classes (via the include parameter, see the documentation). Why are there negative lengths?
8. Print the relative frequencies of the various reservation end types!
9. The results should show that > 98.5% of reservations are either a) released by the user, b) the VM is dropped at the end of the reservation (the user logs in at least once) or c) the user does not acknowledge the spun-up VM and does not fetch the login credentials (such VMs are automatically purged after a while). Everything else is an outlier. Create a data frame that contains only the "majority" classes! Also: drop the cases where the system was not able to serve reservations. From this point on, we will only use this data. (We will refer to this as df1 - you can use another name.) Ideally, we should analyze the outliers, too, but we do not have time for this now.
10. Ensure that no NaN or NaT remains in the data!

6.2.2 Descriptive statistics

11. Print the descriptive statistics for df1; now also interpret the results!
12. Visualize the relative frequencies of the reservations of the different VM types! Interpret the results!
13. Make a histogram of the reservation lengths! Why is it unusable?
14. Create a filtered version of df1 without the reservation length outliers! (we will call it df2, include argumentation for the cutoff point.)
15. On df2: visualize the distributions of the reservation lengths for all VM types that had at least 5 reservations! How can you categorize the VMs based on their reservation length distributions?

6.2.3 Data exploration: capacity utilization

16. On df1: create the time-series a) cluster-wide memory reservation percentage and b) cpu core reservation percentage. (I.e. at time instance t what percentage of the cluster core set / cluster memory is dedicated to a reservation.) You are advised to follow a sampling-based approach with a sensible period time (strike a balance between computational constraints and sampling frequency). Note that pandas has facilities for creating datetime range series. You can assume that

there is no overprovisioning in the cluster and that each host is equipped with 16 cores and 32GB of RAM.

17. Visualize and interpret the distribution of the (sampled) cluster-wide memory utilization and core utilization.
18. Visualize the time series of the utilizations! How would you characterize the workload of the system based on these visualizations?
19. What are the spikes? To answer this question, modify the time series and the time series visualizations so that the contributions of the individual VM types can be differentiated (you are advised to aggregate VM types with insignificant numbers of reservations into a single 'other' category).
20. On df1: compute, visualize and analyze the memory and core reservation percentages for the individual hosts! (You do not have to distinguish VM types at this point.) What can be said about the reservation scheduler of the cluster based on your results?

6.2.4 Spark

21. Get the file platform.csv and upload it in the "My Data" sub-service.
22. In a production setting, we would just point Spark / SparkSQL to an (already stored) big data input spread out on multiple nodes and instruct it to treat it as the data we define our computation steps on. In the lab, we will define our input data in a roundabout way: we parse it into a pandas data frame and create a Spark DataFrame from that. You are advised to the following approach:
```
sqlContext = SQLContext(sc)
panda = pandas.read_csv('/resources/data/platform.csv')
sparkdf = sqlContext.createDataFrame(panda)
```
This is not exactly nice, but works for a lab excercise.
23. Print the first 10 rows of the data frame!
24. Print, inspect and interpret the schema and basic statistics of the data frame! You will need the printSchema and the describe functions; however, for the string columns you will have to do your own scripting.
25. The data frame is in „long format”; ideally, we would like to see the various variable values as columns. Using e.g. pivoting, transform the data into „wide format”!
26. Compute, visualize and interpret the (linear) corrrelation between CPU and memory usage for the cluster as well as the individual machines!
27. Compute, visualize and interpret the (linear) corrrelation between the actual CPU/memory usage of the cluster and the reservation percentages computed in the previous section! Based on your results - what recommendations could you make with respect to capacity planning and/or cluster overprovisioning?