Towards Exploratory Data Analysis for Pharo

Oleksandr Zaytsev
olk.zaytsev@gmail.com

Abstract

Data analysis and visualizations techniques (such as split-apply-combine) make extensive use of associative tabular data-structures that are cumbersome to use with common aggregation APIs (for arrays, lists or dictionaries). In these cases a fluent API for querying associative tabular data (like the ones provided by Pandas, Mathematica or LINQ) is more appropriate for dynamic environments such as Pharo. Despite the fact that many important tools for data-analysis are already implemented in PolyMath, we are still missing.
The simplicity and power of Smalltalk combined with a live environment provided by Pharo creates a perfect environment for data analysis. The advanced debugging and inspecting tools together with the library for agile visualizations allow us to communicate and play with every object in our system. This includes all the logical components of both data and the algorithm. Provided the proper tools and open source libraries for machine learning, statistics, and optimization, Pharo can become both powerful tool for professional data analysts, and a simple environment for everyone who wants to play with a data set. Many important tools and algorithms are already implemented in PolyMath. But the essential part of data analysis toolkit is missing - the specialized data structures for tabular data sets with a simple and powerful API for summarizing, cleaning, and manipulating the data. In this paper I introduce `DataFrame` and `DataSeries` - the new collections specifically designed for working with structured data. I demonstrate how these tools can be used for descriptive statistics and the exploratory data analysis - the critical first step of data analysis which allows us to get the summary of a data set, detect mistakes, determine the relations, and select the appropriate model for further confirmatory analysis.

1 Introduction

All pictures in this paper are the built-in DataFrame visualizations that are created with Roassal2 and can be reproduced by following the steps in section

Paper structure In the first section I provide a brief introduction into the explanatory data analysis. What is it good for? How to do it right? I will also provide you with all the basic knowledge about statistics and data analysis, such as statistical variable, types of variables etc. In the second section I briefly describe the DataFrame - new data structure for data analysis. In the following section I will give a step-by-step example of how to perform EDA on the well-known Iris data set.

2 Exploratory Data Analysis

Exploratory data analysis (EDA) is an approach to analyzing data sets to summarize their main characteristics. It allows us to make some sense of the data by visualizing it and exploring its statistical properties. According to Howard J. Seltmanan, any method of looking at data that does not include formal statistical modeling and inference falls under the term exploratory data analysis [2]. It is an important first step of data analysis which helps us to select the model that we will be fitting to the data during the following steps of confirmatory data analysis.
EDA can be particularly useful for uncovering underlying structure of a dataset, detecting outliers and anomalies, determining relationships among the explanatory variables, assessing the direction and rough size of relationships between explanatory and outcome variables, and selecting applopriate models for further analysis.
By the number of variables and the ... the techniques of exploratory data analysis can be cross-classified into four types: univariate graphical, univariate non-graphical, multivariate graphical, and multivariate non-graphical.

3 DataSeries and DataFrame

DataSeries and DataFrame are the high-level data structures that make data analysis in Pharo fast and easy. [1]
DataFrame can be loaded into a Pharo image with the following Metacello script:

```
Metacello new
```

```
baseline: 'DataFrame';
repository: 'github://PolyMathOrg/DataFrame';
load.
```

DataSeries is an ordered collection that combines the properties of both Dictionary and Array, together with some extra functionality. It has a name and contains an array of data mapped to the corresponding array of keys (index values).
The easiest way of creating a series is by converting it from an array.

```
series := #(a b c) asDataSeries.
```

The keys will be automatically set to the numeric sequence which can be described as an interval (1 to: n), where n is the size of array, and the name will remain empty. Both name and the keys of a DataSeries can be changed later:

```
series name: 'letters'.
series keys: #(a1 a2 a3).
```

DataFrame is a tabular data structure and an ordered collection of columns. It works like a spreadsheet or a relational database with one row per subject and one column for each subject identifier, outcome variable, and explanatory variable. DataFrame has both row and column indices which can be changed if needed. The important feature of a DataFrame is that whenever we ask for a specific row or column, it responds with a DataSeries object that preserves the same indexing.
A simple DataFrame can be created from an array of rows or columns.

```
df := DataFrame rows: #((John 25 true)(Jane 21 false)).
df := DataFrame columns: #((John Jane)(25 21)(true false)).
```

Those two line produce exactly the same DataFrame. Once again, the names (key values) of both rows and columns were not explicitly specified, so by default they will set to numerical indices: #(1 2) for rows and #(1 2 3) for columns. These names can be changed later:

```
df columnNames: #(Name Age IsMarried).
```

4 Exploring Iris Data Set

We can start by loading iris data set from a CSV file.

```
data := DataFrame fromCsv: '/path/to/iris.csv'.
```

DataFrame comes with a built-in collection of data sets that are widely used as examples for data analysis and machine learning problems. Iris is among them, so an alternative way of loading it would be simply

```
data := DataFrame loadIris.
```

Now let's take a look at the first and the last 5 entries in our table. These slices are called *head* and *tail* of a data frame.

```
data head.
data tail.
```

The output will be the following table

```
1  (5.1 3.5 1.4 0.2 #setosa)
2  (4.9 3 1.4 0.2 #setosa)
3  (4.7 3.2 1.3 0.2 #setosa)
4  (4.6 3.1 1.5 0.2 #setosa)
5  (5 3.6 1.4 0.2 #setosa)
```

4.1 Univariate non-graphical EDA

To access a single variable we ask a data frame for a specific column, using its name or number. The result will be a DataSeries object.

```
series := data column: #sepal_width.
series := data columnAt: 1.
```

What we can do with a column depends on a type of statistical variable it represents. The best univariate non-graphical EDA for categorical data is a simple tabulation of the frequency of each category [2]. If the data is quantitative, we can ... `min, max, range, average, median, mode, stdev, variance`

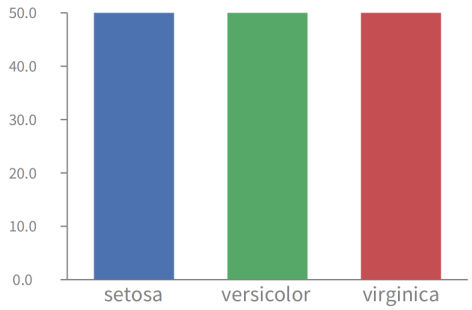
```
series average.
series stdev.
```

4.2 Univariate graphical EDA

4.2.1 Categorical

Histogram is the only graphical technique that can be used for a categorical variable.

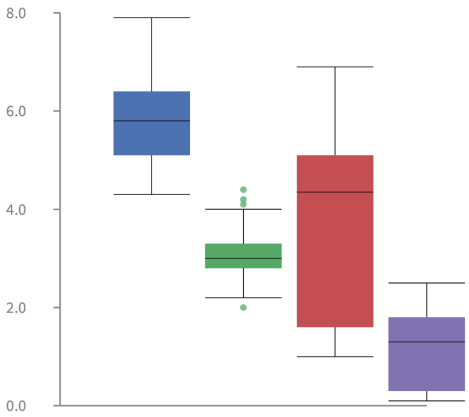
```
var := data column: #species.
var histogram.
```



4.3 Multivariate non-graphical EDA

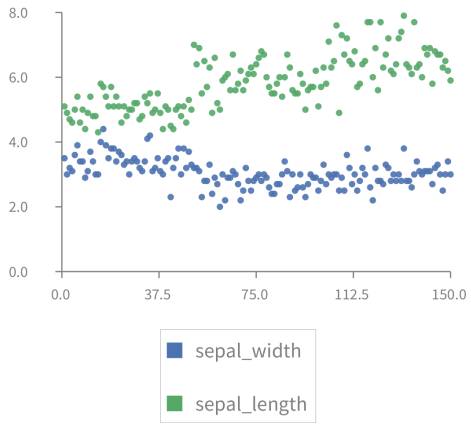
Multivariate non-graphical EDA shows the relationship between two variables in form of either cross-tabulation (categorical data) or statistics (quantitative data).

4.4 Multivariate graphical EDA



Let's look at the scatterplot of two statistical variables representing the width and length of a sepal. To do that we ask our DataFrame to give us specific columns, in our case, `sepal_width` and `sepal_length`, then we ask these columns (the result will be another DataFrame) to visualize themselves.

```
vars := data columns: #(sepal_width sepal_length).
vars scatterplot.
```



5 Future work

At the time of writing this paper DataFrame is capable of... However, a lot of functionality is still missing. For example, we need tools for

- data wrangling
- data aggregation and grouping

6 Conclusion

References

[1] W. McKinney. *Python for Data Analysis*. O'Reilly Media Inc., 2012.

[2] H. J. Seltman. *Experimental Design and Analysis*. 2015.