Data Exploration

Related terms:
Algorithms, Data Warehouses, Data Mining, Clustering, Database, Visualization, Set-Data(), Rapidminer

View all Topics >

Data Exploration

Vijay Kotu, Bala Deshpande PhD, in Predictive Analytics and Data Mining, 2015

Data exploration, also known as exploratory data analysis (EDA), provides a set of simple tools to obtain some basic understanding of the data. The results of data exploration can be extremely powerful in grasping the structure of the data, the distribution of the values, and the presence of extreme values and interrelationships within the data set. Data exploration also provides guidance on applying the right kind of further statistical and data mining treatment to the data. Data exploration tools are a part of standard data analysis software packages from the ubiquitous Microsoft Excel® to advanced data mining software like R, RapidMiner, SAS, IBM SPSS etc. Simple pivot table functions, computing statistics like mean and deviation, and plotting data as a line, bar, and scatter charts are part of data exploration techniques that are used in everyday business setting.

Data Exploration

Vijay Kotu, Bala Deshpande, in Data Science (Second Edition), 2019

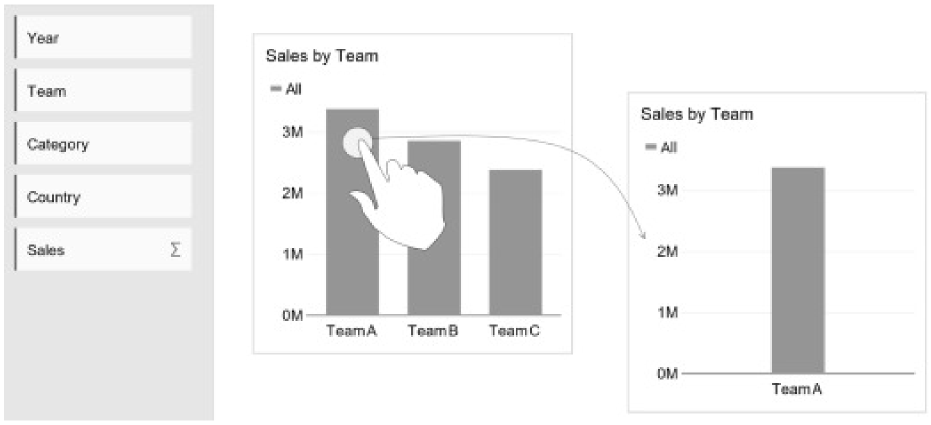
Abstract

Data exploration, also known as exploratory data analysis, provides a set of simple tools to achieve a basic understanding of a dataset. The results of data exploration can be extremely useful in grasping the structure of the data, the distribution of the values, presence of extreme values, and interrelationships within the dataset. Descriptive statistics is the process of condensing key characteristics of the dataset into simple numeric metrics. Some of the common metrics used are mean, standard deviation, and correlation. Visualization is the process of projecting the data, or parts of it, into Cartesian space or into abstract images. In the data science process, data exploration is leveraged in many different steps including preprocessing, modeling, and interpretation of the results.

Visual analytics for software engineering data

Zhitao Hou, ... Dongmei Zhang, in Perspectives on Data Science for Software Engineering, 2016

In MetroEyes, users can perform data exploration through direct manipulation of the visual objects. Say the users want to explore the sales of TeamA. As illustrated in Fig. 1, users can directly select the TeamA bar from the bar chart representing the contribution of each team to Sales, and drag and drop it into the canvas. The tool can then extract from the data source the app sales data contributed by TeamA, and display it in a new bar chart. This data operation selects a dimension value (Team = TeamA) and finds out its app sales, which is equivalent to the SQL query: SELECT Sales, Team FROM AppSales WHERE Team = "TeamA." Note that each bar in the bar chart is a visual object, which can be touched and moved around. Furthermore, each bar represents the percentage of sales each team contributes (eg, the TeamA bar indicates the percentage of sales of TeamA). The visual operations over the object have semantic meanings and correspond to certain data operations.



Sign in to download full-size image

Fig. 1. Explore the sales of TeamA.

Selecting Best Features for Predicting Bank Loan Default

Zahra Yazdani, ... Babak Teimourpour, in Data Mining Applications with R, 2014

8.4 Data Exploration and Preparation

Data mining is closely related to exploratory data analysis. Data exploration is essential to mine data. It is not possible to achieve effective results by mining data without having enough knowledge about the data. Before using data, we must know more about it. It is necessary to analyze the data before applying data mining methods. There are some data visualization tools that facilitate this. We present some of these techniques that were applied to explore our data.

In order to get the clean data, we made use of different preprocessing techniques to have a flawless data set. Data cleaning methods attempt to fill in missing values, smooth out noise, and correct inconsistencies in the data (Han and Kamber, 2006).

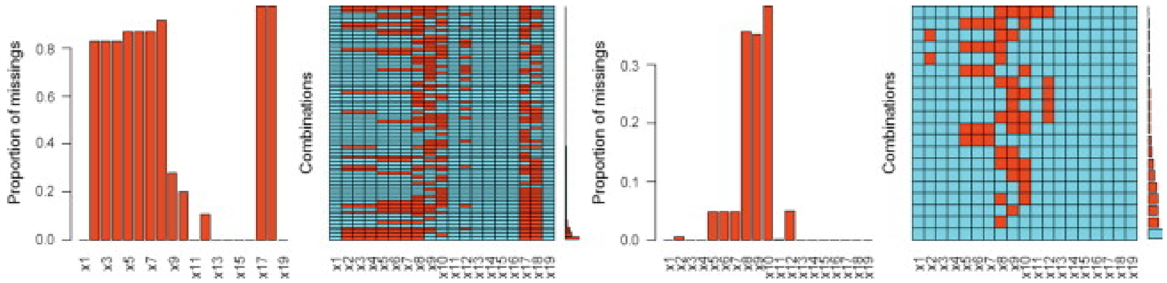
8.4.1 Null Value Detection

Handling missing values as a part of the preprocessing stage is one of the most important aspects in data mining. Without having a clear data set, achieving an effective result is very difficult and sometimes impossible. To have an almost perfect data set, we used several techniques to fill in the missed values.

The exploration of the data set begins using aggr() function of package VIM (Lang et al., 2012). This package introduces new tools for the visualization of missing or imputed values in R, which can be used for exploring the data and the structure of the missing or imputed values.

The aggr() function calculates or plots the number of missing/imputed values in each feature and the number of missing/imputed values in certain combinations of features. It represents many missed values in a data set.

Unfortunately, there was no possibility to estimate all those missed values. The structural characteristics are very important to predict PD, and hence, we had to delete some observations that did not have enough information about the corporate structure. This resulted in a smaller data set. We missed many observations but the remaining ones were so much more complete and efficient. There were also missing values in some behavioral features by structure because the customers were new. To achieve higher accuracy, we divided the data set into two separate data sets and analyzed each set independently. In this chapter, we represent the result of analyzing the one with behavioral features. This data set contains around 1500 observations. Figure 8.1 displays the distribution of null values in the dataset before and after eliminating objects and dividing the data set.



Sign in to download full-size image

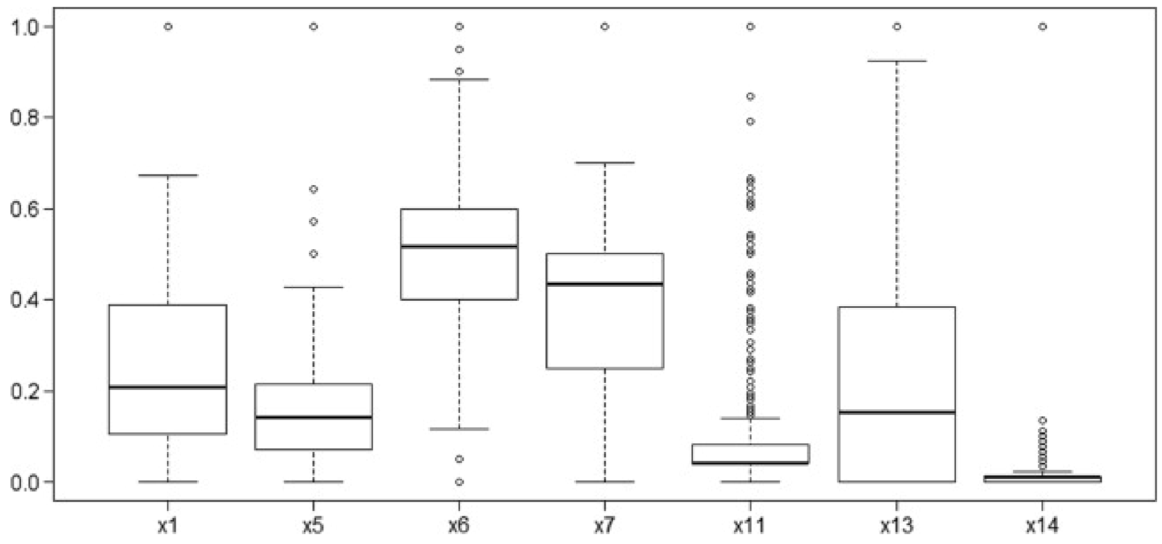
Figure 8.1. Null values in data set before and after eliminating objects and dividing the dataset.

8.4.2 Outlier Detection

Another interesting aspect, albeit a little tricky one in data mining, is the *outlier*. The outlier is a kind of observation, which might deflect the result. A database may contain objects that involve features having very distant values from others. Because the data set contains different types of data, it is wise to investigate the data depending on its type. We used two methods to identify outliers: univariate and multivariate. In the former, we checked each feature separately. Following are some examples of outlier detection in nominal features:

```
> levels(as.factor(mydata[,"x2"]))
[1] "0" "1" "2" "3" "8"
> levels(as.factor(mydata[,"x4"]))
[1] "0" "1" "2" "3" "4" "9" "11" "12"
```

The feature x2 must include values: 1,2,3 and the feature x4 must contains values:1,2,3,9,11,12. So we changed the incorrect values to null. Binary variables were checked too. For numeric features, we applied the boxplot technique. Boxplot is a simple univariate technique for finding outliers and presenting distribution of data. Figure 8.2 displays boxplots for features. To display all the boxplots in one figure, the values were transformed into the [0,1] domain. To achieve clear results, we applied the boxplot in log scale for ratio features. It makes the distribution of features more homogeneous.



Sign in to download full-size image

Figure 8.2. Boxplot for numeric features.

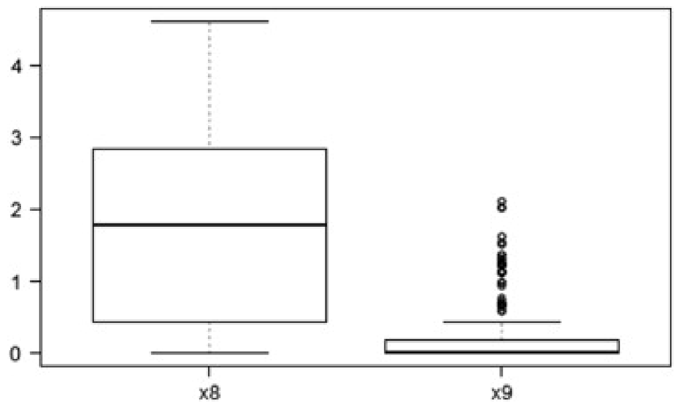
```
# normalization function
norm01 <- function(data,x)
{
```

```
For (j in x)

 {

 data [!(is.na(data[,j])),j]=

 (data[!(is.na(data[,j])),j]-min(data[!(is.na(data[,j])),j])) /

 (max(data[!(is.na(data[,j])),j])-min(data[!(is.na(data[,j])),j]))

 }

return(data)

}
```

Using the normalization function, all features are transformed into the [0,1] scale and it is possible to represent the boxplot in one figure.

```
# drawing boxplot

attach (mydata)

c <- c(1,5,6,7,11,13,14)

data_norm <- norm01(mydata,c)

boxplot(data_norm[,c])

c <- c(8,9,17,18)

boxplot(log(mydata[,c]))
```

Figure 8.3 displays the boxplot for ratio features. Hence, the outliers were identified and their values were replaced with null.



Sign in to download full-size image

Figure 8.3. Boxplot for ratio features in log scale.

We replaced the value of outliers with null to fill them by an efficient imputation technique in the last step.

One of the multivariate methods for detecting outliers is the distance-based method; some of the related algorithms are Fazzy clustering, *SOM*, and Agglomerative *Hierarchical Clustering*. We used the agglomerative hierarchical algorithm because the computation is less complex and it is rather easy to understand. It is a type of clustering technique. In this algorithm, each observation is assumed to be a cluster and in each step, the observations are grouped based on the distance between them (Tan et al., 2005). Each observation that is assigned later has a lower rank. It is seen that the observations with lower rank are outliers because there are dissimilarities between them and the other observations (Torgo, 2011). Therefore, they join into the groups in later stages. For identifying outlier observations, we used the boxplot and the observations with rank under the lower limit were disregarded as outlier data.

Clustering algorithms need distance matrix, and hence, we used the daisy() function of package cluster (Maechler, 2012). As the data set contains mixed types of data the *gower* metric was used to compute distance between objects. The following code is used for outlier ranking:

```
require(cluster)

attach(mydata)
```

```
# creating proximity matrix

dist=daisy(mydata[,-19],stand=TRUE,metric=c("gower"),

 type = list(interval=c(1,5,6,7,11,13,14),ratio=c(7,8,17,18),

 nominal=c(2,12),binary=c(3,4,10,15,16)))

# clustring objects by agglomerative hierarchical clustering

# this function obtains a score for each object

require(DMwR)

outl=outliers.ranking(dist,test.data=NULL,method="sizeDiff", meth = "ward")
```

After ranking, the objects, whose scores are out of range, are disregarded.

Advanced Algorithms for Data Mining

Robert Nisbet Ph.D., ... Ken Yale D.D.S., J.D., in Handbook of Statistical Analysis and Data Mining Applications (Second Edition), 2018

Building Trees Interactively

Building trees interactively has proved popular in applied research, and data exploration is based on experts' knowledge about the domain or area under investigation and relies on interactive choices (for how to grow the tree) by such experts to arrive at "good" (valid) models for prediction or predictive classification. In other words, instead of building trees automatically, using sophisticated algorithms for choosing good predictors and splits (for growing the branches of the tree), a user may want to determine manually which variables to include in the tree and how to split those variables to create the branches of the tree. This enables the user to experiment with different variables and scenarios and ideally to derive a better understanding of the phenomenon under investigation by combining her or his expertise with the analytic capabilities and options for building the tree (see also the next section).

Advanced Algorithms for Data Mining

Robert Nisbet, ... Gary Miner, in Handbook of Statistical Analysis and Data Mining Applications, 2009

Building Trees Interactively

Building trees interactively has proven popular in applied research, and data exploration is based on experts' knowledge about the domain or area under investigation, and relies on interactive choices (for how to grow the tree) by such experts to arrive at "good" (valid) models for prediction or predictive classification. In other words, instead of building trees automatically, using sophisticated algorithms for choosing good predictors and splits (for growing the branches of the tree), a user may want to determine manually which variables to include in the tree and how to split those variables to create the branches of the tree. This enables the user to experiment with different variables and scenarios and ideally to derive a better understanding of the phenomenon under investigation by combining her or his expertise with the analytic capabilities and options for building the tree (see also the next section).

Snap-Together Visualization: A User Interface for Coordinating Visualizations via Relational Schemata

Chris North, Ben Shneiderman, in The Craft of Information Visualization, 2003

3.3 Combined Analysis

Together, these studies indicate the breakpoint at which time savings during data exploration surpass interface construction time. The 2nd study used the same interface constructed in the I" study. The time cost of constructing the interface was 2–5 minutes, while it saved 0.5–1.5 minutes over the Detail-Only interface for more difficult tasks. Hence, after a few tasks, users are already reaping savings with snapping their own interface. Of course, it is difficult to factor in learning time and effects of sharing snapped interfaces. Nevertheless, this simple analysis is revealing. Customized information visualization is within the grasp of novice users.

Introduction: overview of academic and professional publishing

Robert Campbell, in Academic and Professional Publishing, 2012

Journals and data

Our current era of research was described by Jim Gray of Microsoft as being all about data exploration, unifying theory + experiment + simulation. It should offer tremendous opportunities for publishers. The journal has been described as the ultimate metadata for a researcher's data yet until recently the relationship between a journal article and the underlying data on which it is based has received little attention. As outlined by Efke Smit: 'data and publications belong together because publications make data discoverable; are the most thorough metadata of data; provide the author/research credits for the data; and gain depth by supplying data' (Smit *et al.*, 2011).

Again we are seeing policy driven by funders who have become more aware of the importance of improving access to the accelerating quantity of data their funding has generated. They have been supported by publishers in this; for example, see the declaration on Open Access to Data by the STM Association and ALPSP (STM/ALPSP websites, 2006).

Currently, funders' policies on data management are not well aligned with market needs, with policies varying by funder, type of research, university and discipline. The situation in publishing is also unclear. For example, in the Parse Insight 2009 survey, 71 per cent of the larger publishers stated that authors can submit their underlying digital research data yet 69 per cent of the larger publishers said they have no preservation arrangements for digital research data (Knipers and van der Hoeven, 2009). The accessibility of data lags behind the accessibility of journals. In the 2010 PRC survey only 38 per cent of researchers found it easy or fairly easy to access data sets and data models.

It seems likely that libraries and their repositories will take on much of the responsibility for managing data. Funders will require a plan for data management in any grant proposal and the researcher will usually go to their library colleagues to provide for and then implement this. Where librarians once employed subject specialists they will now recruit data managers. And we are seeing Oldenburg's four functions applied to data management. We could even see the later 'fifth function' – i.e. generating citation data used for measuring impact – operate for data as researchers will need to demonstrate the impact of their work. DataCite, a library-led organisation, is co-operating with CrossRef to provide DOIs for linking, which could create the necessary basis for research data metrics. A new community of data managers could forge a new and productive relationship with journal publishers, but it is early days. Alongside all the technical issues there is no clear business model.

Integration of Big Data and Data Warehousing

Krish Krishnan, in Data Warehousing in the Age of Big Data, 2013

Lexical processing

This layer can be applied to both input data processing of Big Data and the processing of data exploration queries from the visualization layer. Lexical processing includes processing tokens and streams of text. The three main subcomponents of lexical processing include:

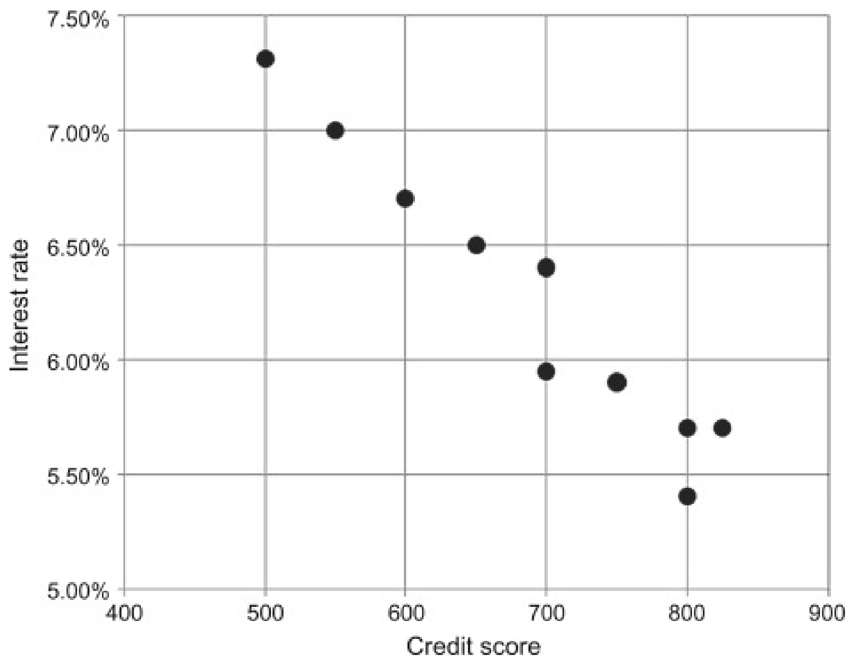
- *Entity extraction*—a process to identify key data tokens that can include keys and master data elements. For example, identifying "product_name" from a twitter feed.
- *Taxonomy*—a process to navigate across domains within the text stream to identify the contexts in the text that may be feasible, and discover relationship attributes for cross-hierarchy navigation.
- *Relationship models*—a process to derive the relationship between different data elements and layers resulting in an exploration roadmap. This process will use outputs from the prior components in this process.

Data Science Process

Vijay Kotu, Bala Deshpande, in Data Science (Second Edition), 2019

2.2.1 Data Exploration

Data preparation starts with an in-depth exploration of the data and gaining a better understanding of the dataset. Data exploration, also known as *exploratory data analysis*, provides a set of simple tools to achieve basic understanding of the data. Data exploration approaches involve computing descriptive statistics and visualization of data. They can expose the structure of the data, the distribution of the values, the presence of extreme values, and highlight the inter-relationships within the dataset. Descriptive statistics like mean, median, mode, standard deviation, and range for each attribute provide an easily readable summary of the key characteristics of the distribution of data. On the other hand, a visual plot of data points provides an instant grasp of all the data points condensed into one chart. Fig. 2.3 shows the scatterplot of credit score vs. loan interest rate and it can be observed that as credit score increases, interest rate decreases.



Sign in to download full-size image

Figure 2.3. Scatterplot for interest rate dataset.