Matrix Explorer: A Web Application for Data Exploration

Anonymous Author(s)
Affiliation
Address
email

Abstract

Exploration and visualization of tabular data is crucial to driving discovery across a multitude of different basic science fields. Unfortunately, given the vast array of various statistical tools that have been developed, finding the proper techniques to explore a novel dataset is more of an art than an exact science. Hence, researchers often choose the wrong tools to explore their data, leading to poor analysis. In this paper we describe Matrix Explorer, a novel data exploration and visualization web application that allows users to analyze uploaded tabular datasets. Within Matrix Explorer we have implemented techniques we believe are relevant to and best practice for any sorts of data analysis, including data visualization via heatmaps, scatter/line plots, marginal distributions, and two-dimensional embedding and data exploration via k-means clustering, dendrograms, correlation matrix computation, and outlier detection. We validated Matrix Explorer by showing that it produces correct results for the iris flower dataset.

1 Introduction

The past decade has seen an explosion in studies generating huge quantities of data. In response, researchers now require tremendous computational power and statistical frameworks to process their data. Unfortunately, exploratory analysis of novel datasets remains more of an art than an exact science, mainly because of the difficulty associated with choosing which of the vast variety of different developed techniques to use. The result of this phenomenon is that the a large quantity of collected data, especially in the basic sciences, is rather poorly analyzed, undoubtedly slowing the pace of new discoveries.

In response, we have developed a framework within which users with minimal statistical knowledge can explore their data. To the best to of our knowledge, such a tool does not yet exist. The R, MATLAB, and Python languages, and associated toolboxes and packages, still require users to have a non-negligable understanding of underlying methods in order to conduct proper data analysis. Other attempts at decreasing the barrier to entry for quality exploratory data analysis such as GGobi, as described in Swayne et al. (2003), and ViSta, as discussed in Valero-Mora and Ledesma (2011), have not resolved the issue, as both still require a significant learning curve to utilize.

In this paper we describe Matrix Explorer (MX), a novel data exploration web application driven by the R Shiny package. Within MX we have included several techniques we have deemed best practices for starting a new exploratory analysis task. MX provides users with minimal experience in statistics the tools necessary to conduct a basic characterization of their data.

Submitted to 29th Conference on Neural Information Processing Systems (NIPS 2016). Do not distribute.

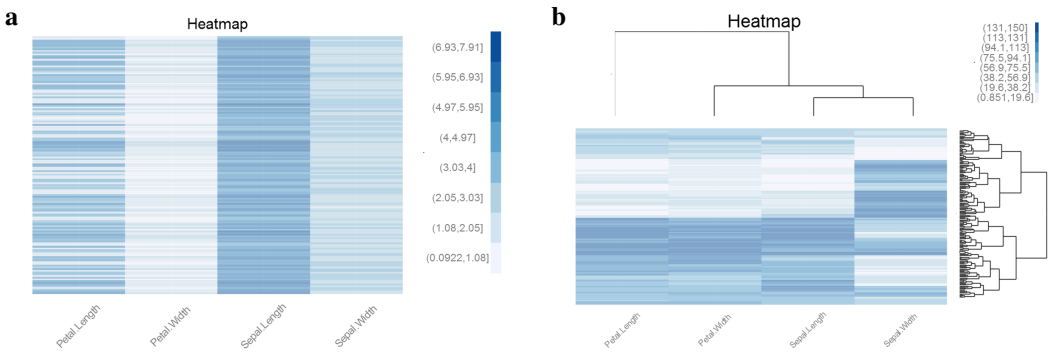

Figure 1: Heatmap of iris dataset for various visualization options. Sample labels are not shown. (a) Heatmap of raw data. Note that there is no evidence for the existence of clusters. (b) Heatmap of data converted to rank with dendrogram and associated row and column sorting. The existence of at least two clusters is clear.

2 Methods & Result Validation

MX is driven by the R Shiny package that allows for the construction of interactive web applications using R as their backend. The Shiny server is hosted on a Docker instance. The functionality of MX is divided into several different submodules, organized as tabs in the web application, described in depth below. We validated the basic data analysis functionality of MX by exploring the iris flower datasets (descriptions and visualizations below).

The iris dataset is composed of 50 samples from each of three types of iris flower (*setosa*, *virginica*, and *versicolor*). For each sample four features are measured, specifically the length and width of the sepal and petals. For the purposes of the validation, we added a fifth feature column with the class label of each sample. We then set out to show that MX can detect samples clustering into three distinct groups.

2.1 Data Upload & Selection

Upon opening the application, the user is first prompted to upload a dataset. Note that MX assumes that rows in the dataset represent samples and columns represent features. Once uploading is complete, the user may then select the rows and columns that he or she wishes to analyze and also flag a specific column as containing class labels. The current implementation accepts comma separated variable (CSV) files and is optimized for sizes below 1 GB. To begin our validation process, we converted our iris dataset to CSV format and uploaded it to MX using the interface.

2.2 Heatmap & Dendrogram

A heatmap provides one of the most minimally-processed ways of visualizing the data, besides simply looking at the data table. However, oftentimes clear trends in the data are obscured on a heatmap because of variations in the range of the data across features. Normalization resolves this issue. The usefulness of heatmaps is further increased by accompanying dendrograms for easy identification of clusters within the data. MX allows users to construct a heatmap of their data and gives them the option to normalize it via conversion to z-scores, quantiles, or ranks. For datasets with less than 100 rows and columns, MX can use hierarchical clustering based on both features and samples to compute two dendrograms for the dataset and display them, reordering the columns and rows as necessary.

We used the heatmap feature of MX to do our initial exploration of the iris dataset. We found that a simple heatmap of the unnormalized dataset, as shown in Figure 1a, is not particularly useful. There is no clear structure to the data. This is where normalization schemes, as well as dendrograms and the associated sorting they impose upon the rows and columns of the heatmap, become useful. A heatmap and dendrograms for the ranked data is shown in Figure 1b. Note that the existence of at least 2 clusters is now apparent.

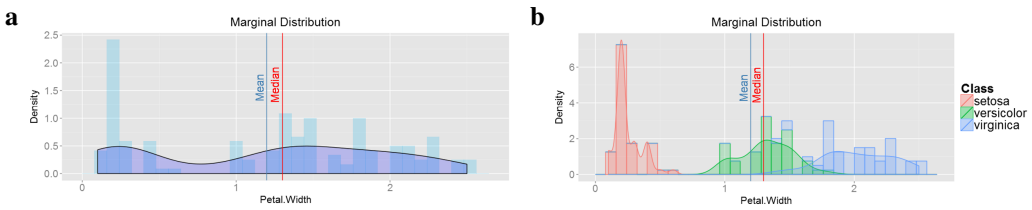

Figure 2: Marginal of the petal width feature of the iris dataset. (a) Histogram and kernel density estimate of the samples without conditioning on class. Note that it is unclear whether the distribution is multi-modal. (b) Histogram and kernel density estimate of the samples after conditioning on class. It is now clear that the marginal distribution is tri-modal.

Figure 3: Robust Mahalanobis distances with choice of alpha cut-off value to reject samples as outliers on the iris dataset. Note that 50 of the 150 samples are flagged as outliers.

2.3 Sample Summary

One of the initial steps in exploring a new dataset is to look at the marginal distributions of the individual feature columns, as this helps direct future work. MX allows users to visualize these marginal distributions by generating a histogram, a kernel density estimate, or an overlay of both. The mean and median is indicated on the plot. Additionally, if the user labeled a feature as a class label column, he or she has the option of conditioning the marginal on classes. This latter feature is incredibly useful for feature selection and determining the separability of the data based on class labels within the feature space.

We used MX to construct the feature marginals. The distribution for petal width is shown in Figure 2a. Note that it is unclear whether the distribution is multi-modal and, if it is, how many modes compromise the distribution. This is where conditioning on classes becomes useful (Figure 2b). After conditioning, it is now clear that the marginal is tri-modal and that the petal width feature does a good job of separating the samples into the three classes.

2.4 Outliers

In order to determine outliers it is necessary to determine how far away a specific sample is from other samples by establishing and computing a distance metric. In this case, robust Mahalanobis distances were used, as described in Hubert et al. (2008). Briefly, a robust estimate of the sample mean and covariance was first approximated via either: 1) the fast minimum covariance determinant algorithm (FAST-MCD), as described in Rousseeuw and Driessen (1999); or 2) the Orthogonalized Gnanadesikan-Kettenring algorithm (OGK), as described in Maronna and Zamar (2002). Since FAST-MCD scales poorly with increasing dimensionality, FAST-MCD was used only for low dimensional datasets, while OGK was used for high dimensional datasets. Next, using these robust measures, the Mahalanobis distance of each sample was computed. As explained by Hardin and Rocke (2012), the squares of the robust Mahalanobis distances are approximately distributed χ^2_n, where n is the dimensionality of the dataset. It is then possible to define an alpha value above which to reject a

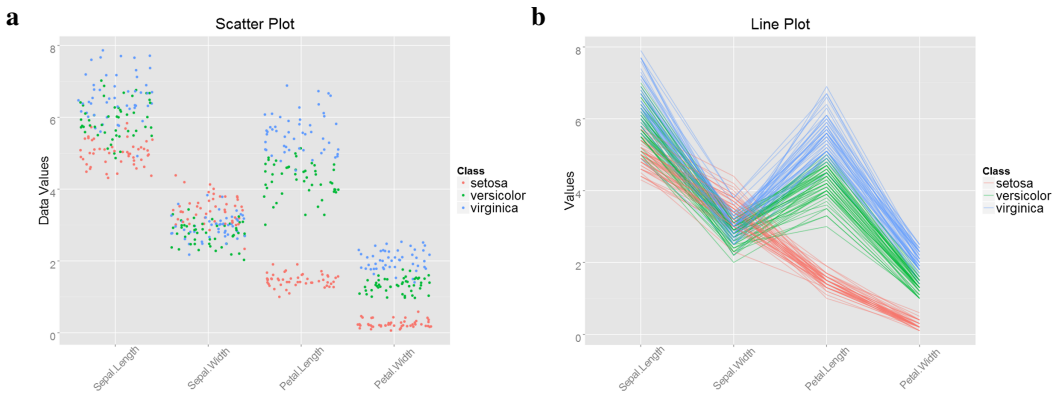

Figure 4: Visualization of the samples across all the features in the iris dataset. (a) Jittered scatter plot of the sample values across the features and colored based on class label. Note that the multi-modal nature of the petal width and petal length marginal distributions is clear. It is also apparent that the samples from the *Iris setosa* class are very different from the samples from the other two classes. (b) Line plot showing the behavior of individual samples across the features. Visually, it is clear how differently the samples from the *Iris setosa* class behave compared to samples from the other classes. Note also that we can now visually see the absence of actual outliers.

sample as an outlier. MX allows the user to set this alpha value and graphically displays the robust Mahalnobis distances and resulting threshold. The robust Mahalanobis distances for the iris data are shown in Figure 3. Based on the analysis, MX labels 50 of the 150 samples as outliers. This is a very high fraction, and it is quite unlikely that all of these are actually outliers. It is more likely that the samples flagged as outliers actually belong to a single class of iris flower that simply is very far away from the other two classes. When we examined the class labels, this is exactly what we found: all 50 of these samples belong to the *Iris setosa* class, samples from which are, as a whole, apparently quite far from the samples belonging to the *Iris virginica* and *Iris versicolor* classes. Interestingly, the outlier analysis did not indicate a difference in distance between the samples from the *Iris virginica* and *Iris versicolor* classes.

2.5 Correlation

Understanding correlations within a dataset is highly relevant, beyond simply checking assumptions and selecting features to build classifiers. Correlations aid in providing an intuitive understanding of the underlying patterns within the data. For example, in gene expression datasets correlations may indicate the biological importance of various genetic markers (for example, see Shi and Horvath (2012)). MX computes the Pearson's correlation amongst features and displays it to the user. To better understand the interactions and distributions of features, MX can also compute the euclidean distance between features. To account for scaling effects, users have the option of pre-processing the data via conversion to z-scores, quantiles, or ranks. Furthermore, users can remove the outliers detected in the outlier tab from the data before computing the metrics. Using these feature on the iris data, we found that all 4 of the features are strongly correlated with one another (data not shown).

2.6 Feature Summary

MX allows users to visualize the marginal distributions of all the features on the same plot. This is done via a jittered scatter plot, a line plot (where each line represents the values a sample takes on at the corresponding feature), a vector of sample means with standard error bars, a box plot, and a violin plot. In order to ensure that outliers do not skew the visualizations, users have the option of removing outliers detected in the outlier tab from the dataset before generating the plots.

A jittered scatter plot of the iris data, colored by class label, is shown in Figure 4a. The multi-modal nature of the petal length and width features is clear from the plot. Furthermore, the two petal-related features seem to be optimal for separating the data into the three classes. Consistent with the outlier analysis, it is apparent here that the samples from the *Iris setosa* class differ strongly from the samples

Figure 5: Two-dimensional embedding of samples using PCA followed by coloring using labels given by k-means clustering with $k = 3$ using (a) raw data and (b) z-score normalized data. Note that the k-means result is significantly worse in (b) compared to (a), to the point that it obfuscates the obvious existence of clear clusters.

from the *Iris virginica* and *Iris versicolor* classes. A line plot colored by class label of the same data is shown in Figure 4b. Again, we see how similar the samples from the *Iris virginica* and *Iris versicolor* classes are to each other and how different they are from the samples from the *Iris setosa* class. Interestingly, we can now visually see that there are no obvious outliers in the data. These observations support the idea that the samples flagged as outliers by MX were not "real" outliers, but instead reflected a distinct class.

2.7 Embedding & Clustering

One of the most intuitive ways of visualizing bivariate data is a scatter plot. Although this is less straightforward for data with more than two dimensions, various dimensionality reduction techniques can be used to embed the data within a two-dimensional coordinate system for simple examination of general structure and clustering. Dimensionality reduction techniques can be further reduced into linear and non-linear methods, both of which have their own unique advantages and disadvantages (for an overview see Van Der Maaten et al. (2009)). Within MX we have implemented one linear method – principle component analysis (PCA) – and one non-linear method – t-distributed stochastic neighbor embedding (tSNE). For a thorough description of how tSNE works, see Van der Maaten and Hinton (2008).

PCA is extremely sensitive to: (1) the scaling of the data, and (2) the presence of outliers. Hence, to overcome (1), MX allows users to normalize the feature columns by converting either to z-scores, quantiles, or ranks. Note that conversion to quantiles or ranks also decreases the sensitivity of the embedding to outliers, hence aiding in overcoming (2). MX also allows users to resolve (2) more directly by giving them the option to remove the outliers detected in the outliers tab before generating the visualization. In order to inform the users of how much information was lost by embedding the data set into two dimensions, MX also displays a scree plot, indicating the percentage of retained variance after PCA.

For our validation analysis, we implemented two-dimensional embedding via PCA for the raw (Figure 5a) and z-score normalized (Figure 5b) iris data after coloring using labels attained via k-means with $k = 3$. Based on these visualizations it is that the k-means clustering result for the z-score normalized data is far worse than the result for the raw data, as it does not indicate the existence of two easily separable clusters. This suggests that the current standard of normalizing to z-scores is not alway optimal. As we have seen above, the smaller, more distant cluster is likely composed of samples belonging to the *Iris setosa* class.

3 Conclusion

We have created MX, a web application driven by the R Shiny package, that allows for basic data exploration and visualization of small to medium sized datasets. We have included functionality that reflects what we believe to be best practice when exploring a new dataset. We have validated the functionality of MX by demonstrating its capability to characterize the iris flower dataset, including the identification of three distinct classes.

Although the algorithms implemented within MX are relatively simplistic, they are highly relevant to MXs target user base: scientists without the experience necessary to select the statistical techniques

to use when initially exploring their data. Indeed, the implemented algorithms give such researchers a foundation from which they can further analyze their data using tools beyond the scope of this work. As such, MX helps resolves a very important issue, that impeded progress in many fields of research. It is a significant step towards supporting all researchers in using appropriate statistical tools to explore their datasets.

References

Hardin, J. and D. M. Rocke (2012). The distribution of robust distances. *Journal of Computational and Graphical Statistics*.

Hubert, M., P. J. Rousseeuw, and S. Van Aelst (2008). High-breakdown robust multivariate methods. *Statistical Science*, 92–119.

Maronna, R. A. and R. H. Zamar (2002). Robust estimates of location and dispersion for high-dimensional datasets. *Technometrics*, 307–317.

Rousseeuw, P. J. and K. V. Driessen (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics 41*(3), 212–223.

Shi, T. and S. Horvath (2012). Unsupervised learning with random forest predictors. *Journal of Computational and Graphical Statistics*.

Swayne, D. F., D. T. Lang, A. Buja, and D. Cook (2003). Ggobi: evolving from xgobi into an extensible framework for interactive data visualization. *Computational Statistics & Data Analysis 43*(4), 423–444.

Valero-Mora, P. M. and R. D. Ledesma (2011). Using interactive graphics to teach multivariate data analysis to psychology students. *Journal of Statistics Education 19*(1), 1–19.

Van der Maaten, L. and G. Hinton (2008). Visualizing data using t-sne. *Journal of Machine Learning Research 9*(2579-2605), 85.

Van Der Maaten, L., E. Postma, and J. Van den Herik (2009). Dimensionality reduction: a comparative. *Journal of Machine Learning Research 10*, 66–71.