Data Analysis and Visualization

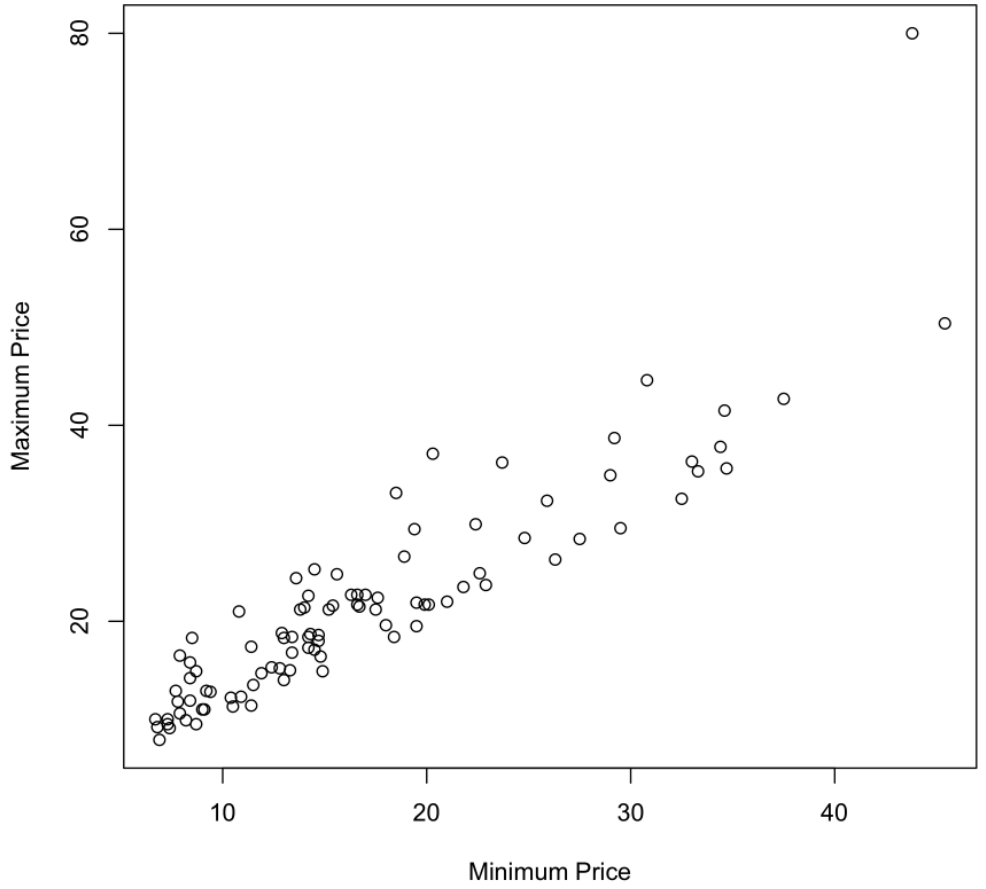
Universität Konstanz

Information Visualization

Scatterplots



Dr. Johannes Fuchs



Scatterplots

	#	Task	Description
object-centric	1	Identify object	Identify the referent from the representation
	2	Locate object	Find a particular object in its new spatialization
	3	Verify object	Reconcile attribute of an object with its spatialization (or other encoding)
	4	Object comparison	Do objects have similar attributes? Are these objects similar in some way?
browsing	5	Explore neighborhood	Explore the properties of objects in a neighborhood
	6	Search for known motif	Find a particular known pattern (cluster, correlation)
	7	Explore data	Look for things that look unusual, global trends
aggregate-level	8	Characterize distribution	Do objects cluster? Part of a manifold? Range of values?
	9	Identify anomalies	Find objects that do not match the 'modal' distribution
	10	Identify correlation	Determine level of correlation
	11	Numerosity comparison	Compare the numerosity/density in different regions of the graph
	12	Understand distances	Understanding a given spatialization (e.g. relative distances)

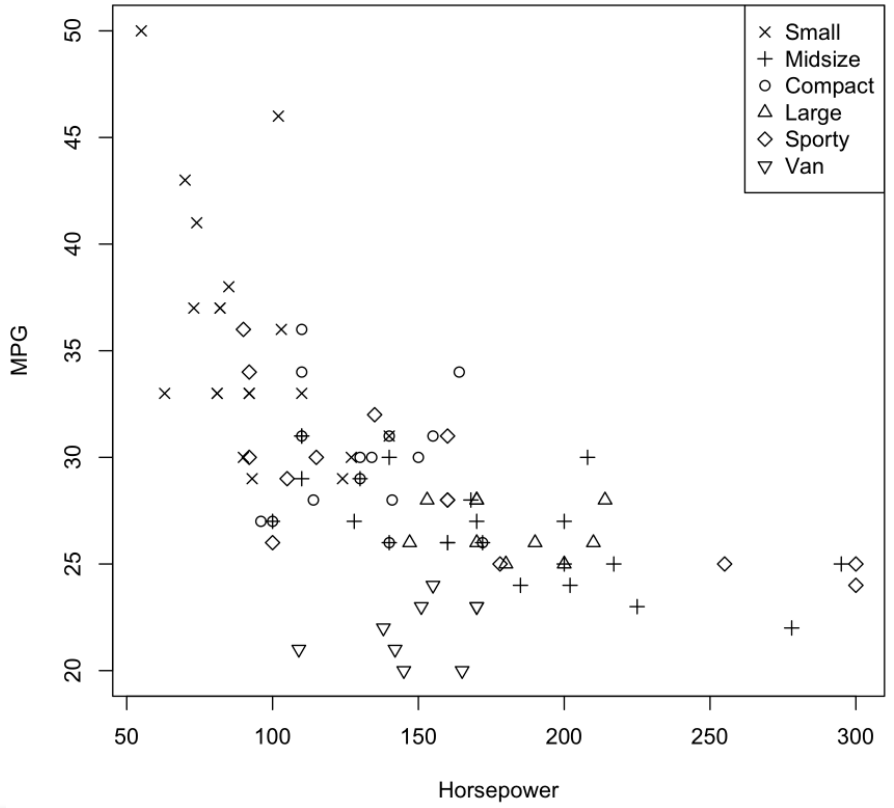
Scatterplots - Tasks

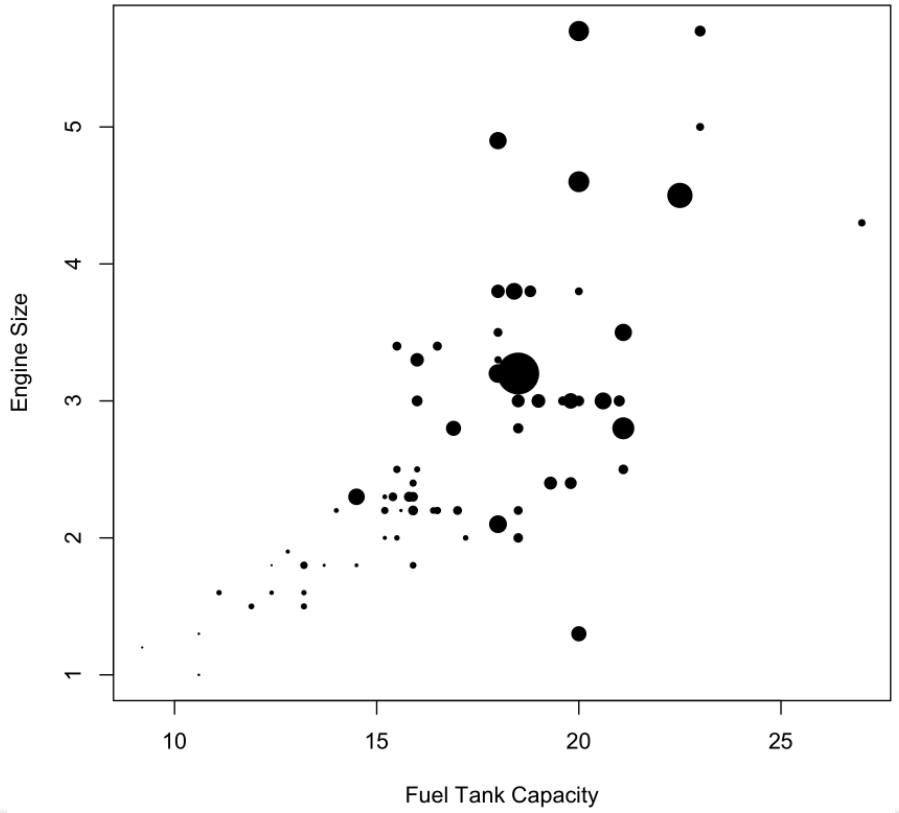
Sarikaya and Gleicher, 2018

- Advantage:
  - Easy to read single data entries
  - Easy to understand patterns

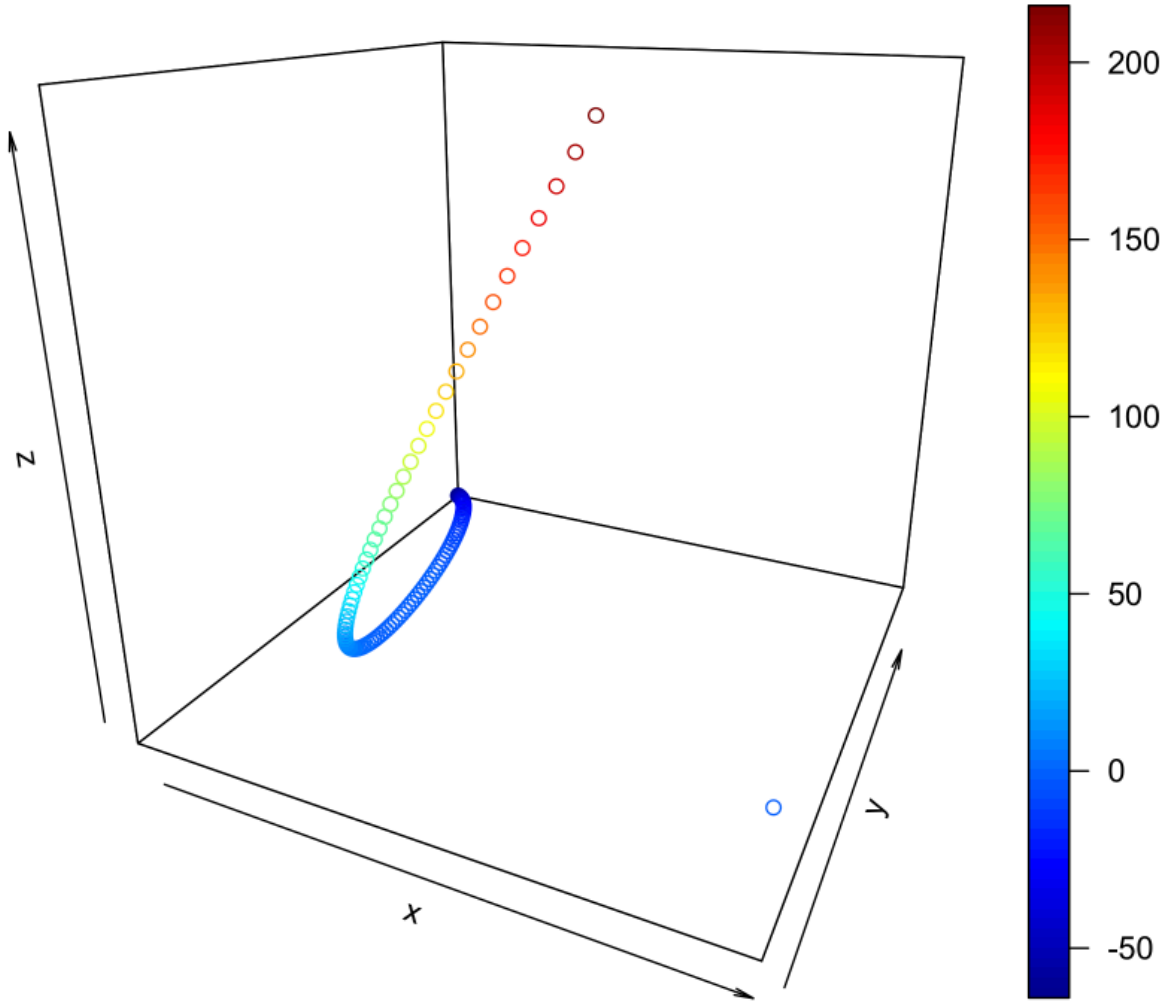
- Disadvantage:
  - Overplotting
  - Limited to two dimensions

Add additional visual variables to the scatterplot



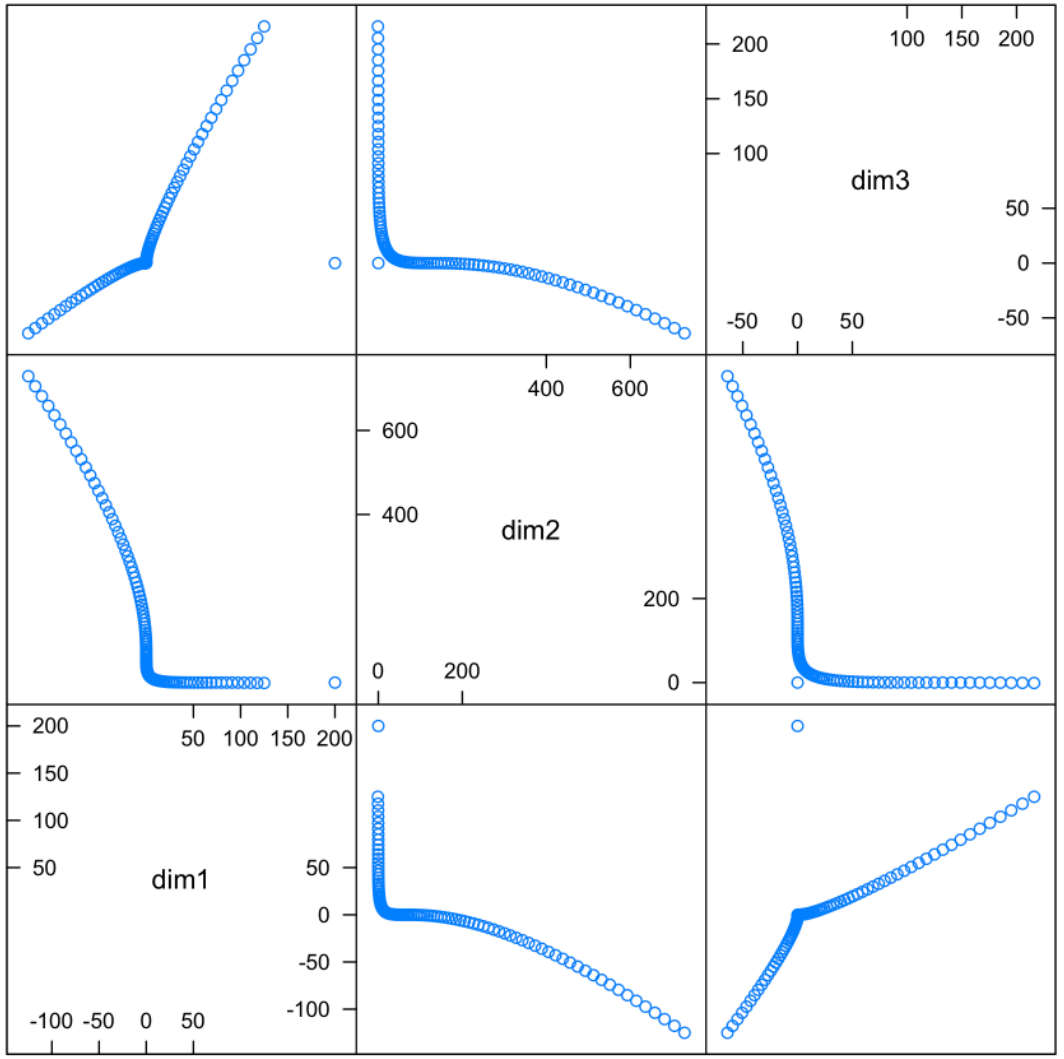


Add another dimension to the scatterplot (3D)



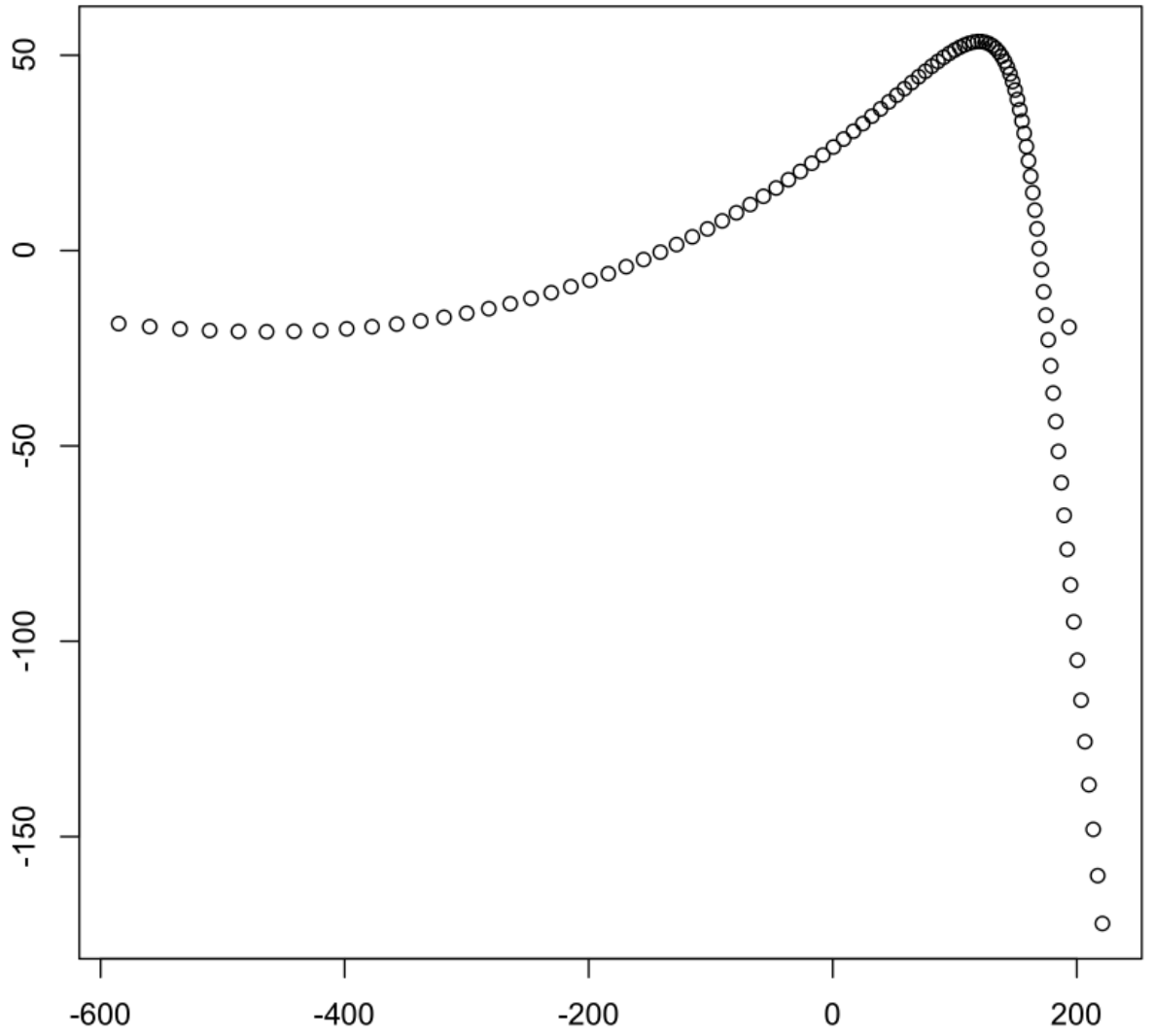
Scatterplots - Limited to Two Dimensions

Display multiple scatterplots



Scatterplots - Limited to Two Dimensions

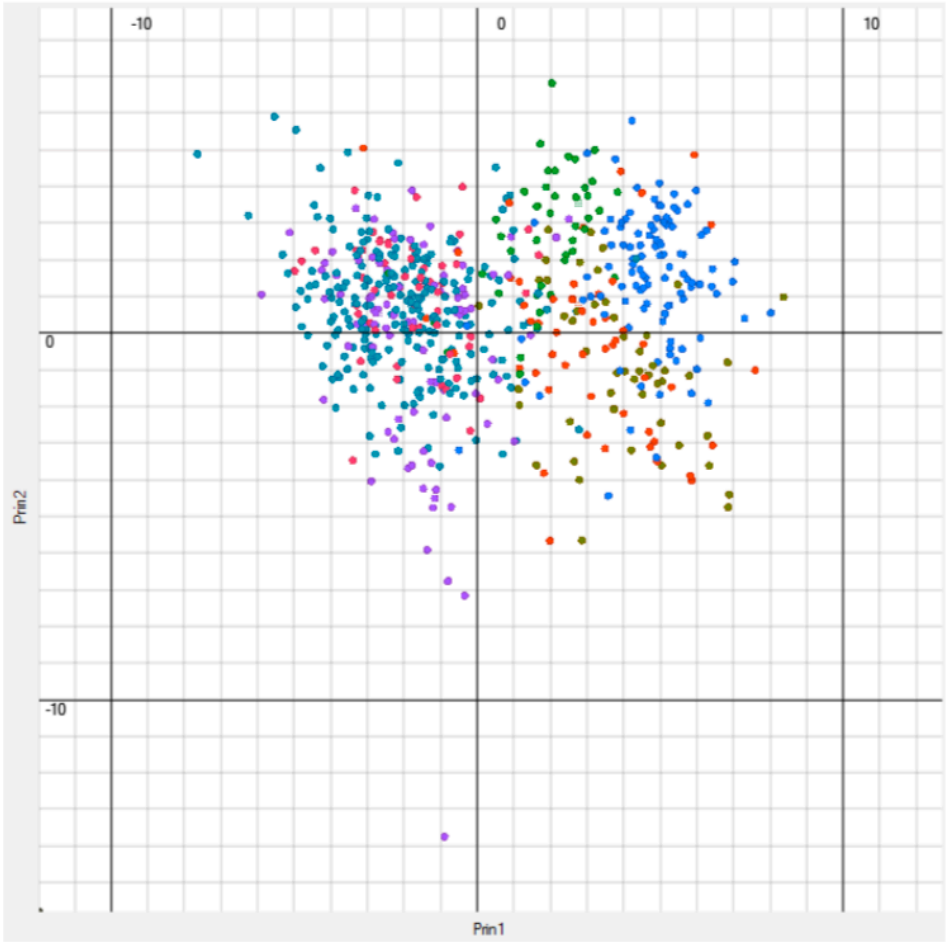
Project to a two dimensional space



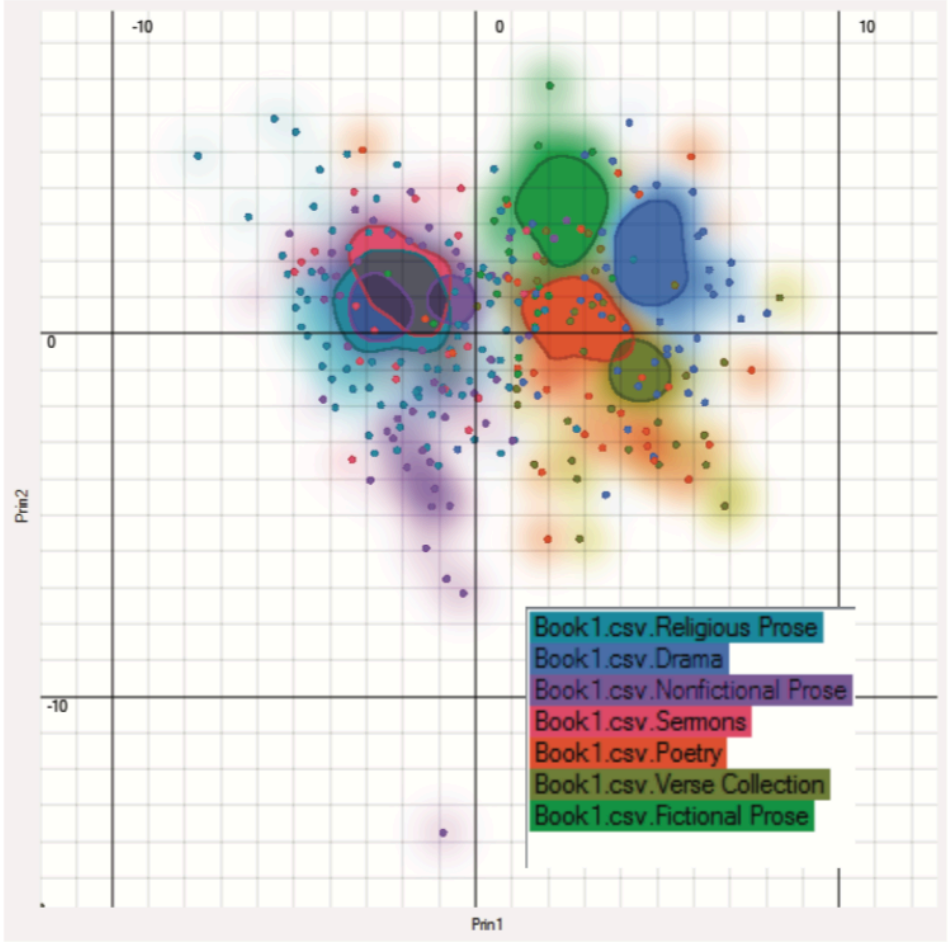
Scatterplots - Limited to Two Dimensions

How to avoid overplotting?

- Sampling
- Filtering
- Clustering
- Binning
- Distortion



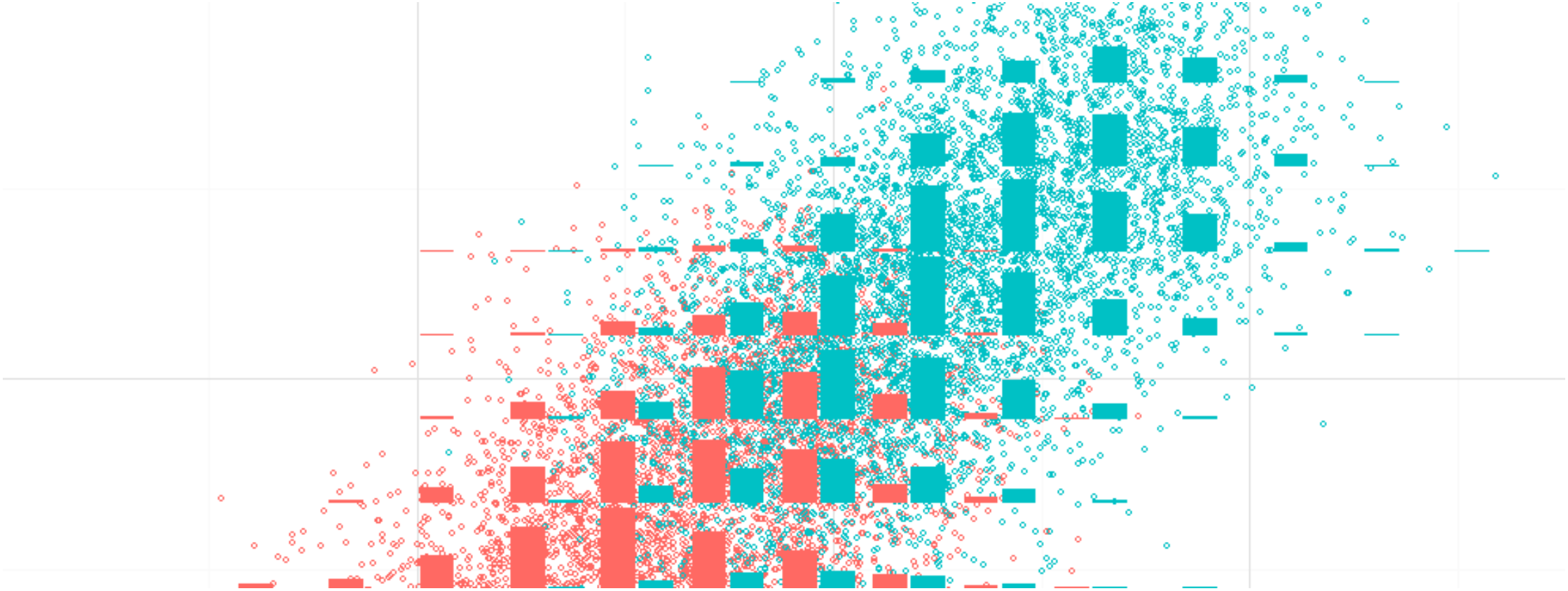
(b) Scatter plot



(a) Splatterplot

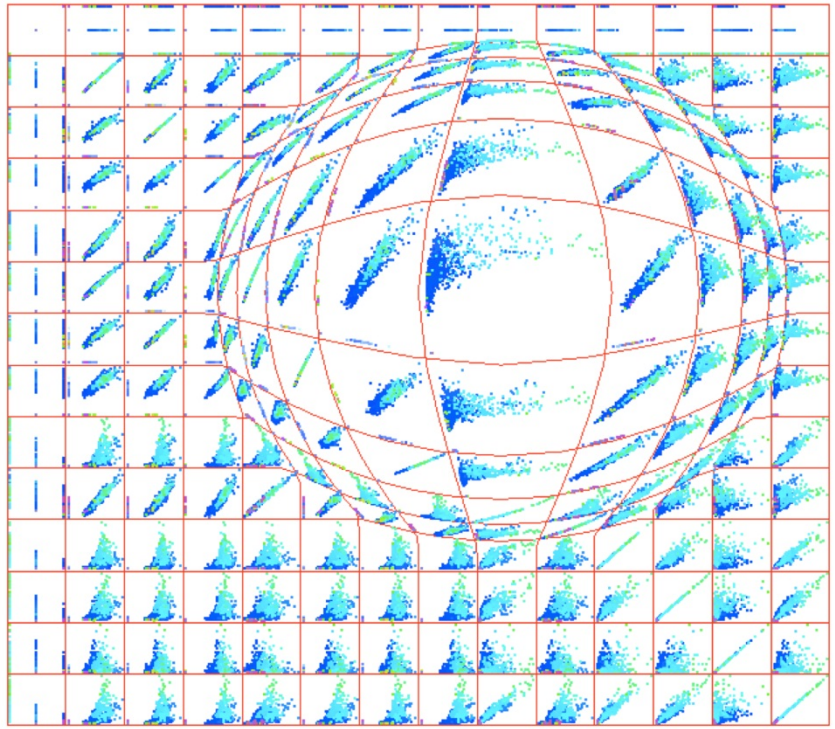
Scatterplots - Clustering

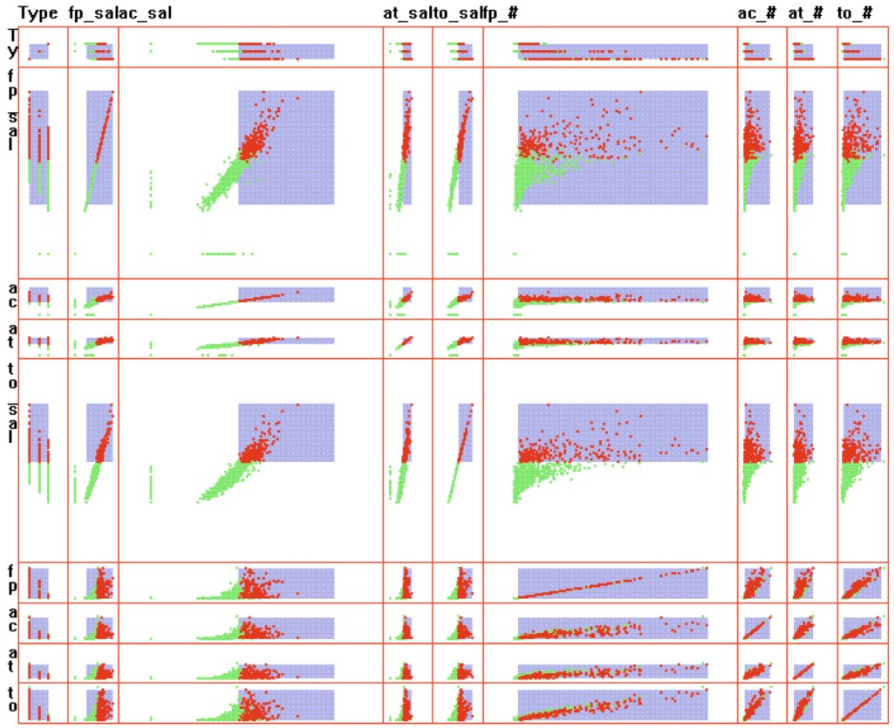
Mayorga and Gleicher, 2013



Scatterplots - Binning

https://shapescience.xyz/blog/reducing-overplotting-in-scatterplots/





Scatterplots - Distortion

Ward and Yang, 2004

D3.js is a JavaScript library for manipulating documents based on data.

- D3 stands for **D**ata **D**riven **D**ocuments
- It utilizes HTML, CSS, SVG
- It combines powerful visualization components and a data-driven approach to DOM manipulation
- Resources: https://github.com/d3/d3/wiki



HTML is a markup language for describing web documents.

- HTML stands for **H**yper **T**ext **M**arkup **L**anguage
- A markup language is a set of markup tags
- HTML documents are described by HTML tags
- Each HTML tag describes different document content

HTML– Introduction (RECAP)

Basically, an HTML document consists of the following:

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4  <title>Page Title</title>
5  </head>
6  <body>
7
8  <h1>My First Heading</h1>
9  <p>My first paragraph.</p>
10
11 </body>
12 </html>
```

The **DOCTYPE** declaration defines the document type to be HTML

The text between **<html>** and **</html>** describes an HTML document

The text between **<head>** and **</head>** provides information about the document

The text between **<title>** and **</title>** provides a title for the document

The text between **<body>** and **</body>** describes the visible page content

The text between **<h1>** and **</h1>** describes a heading

The text between ** ** and ** ** describes a paragraph

SVG Facts:

- Written in XML -> all elements must be properly closed.
- SVGs do not lose any quality if they are zoomed or resized.
- Visual elements must be enclosed by an SVG tag.
- The coordinates (0, 0) are in the upper left corner.

Scalable Vector Graphics (SVG) (RECAP)

SVG Shapes:

- Rectangle `<rect id="rect" x="50" y="20" width="50" height="50" />`
- Circle `<circle id="circle" cx="50" cy="60" r="40" />`
- Ellipse `<ellipse cx="200" cy="80" rx="100" ry="50" />`
- Line `<line x1="0" y1="0" x2="200" y2="200" />`
- Polyline `<polyline points="20,20 40,25 60,40" />`
- Polygon `<polygon points="200,10 250,190 160,210" />`
- Path `<path d="M150 0 L75 200 L225 200 Z" />`
- Text `<text x="250" y="250">SVG Text</text>`

SVG – Elements (RECAP)

SVG Groups:

- Group elements do not have an x or y position.
- You have to position the element using the transform attribute.
- The position of all elements within the group is also updated.

```
<g id="rect" transform="translate(100,100)" />
```

SVG – Group-Element (RECAP)

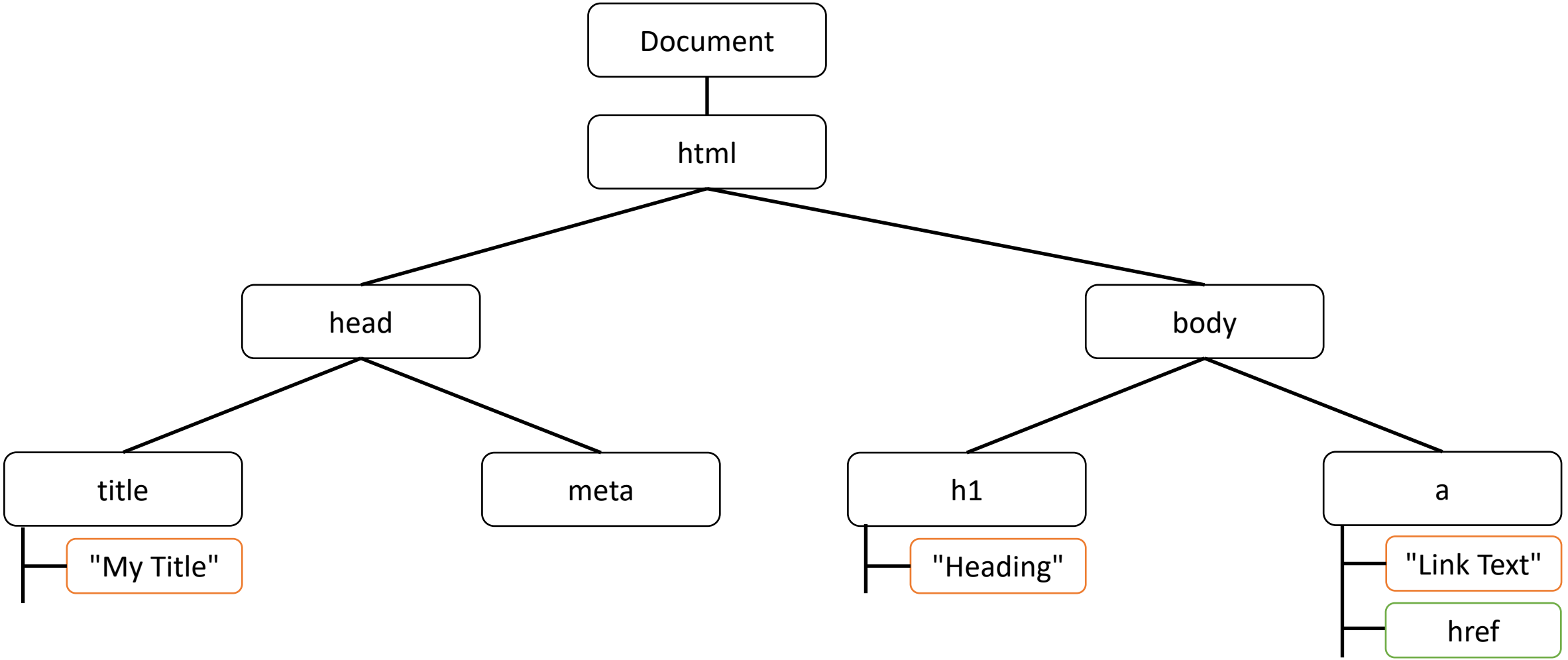
DOM - What is it used for:

- Basically it is the interface between HTML and JavaScript
- A browser parses the HTML document and creates elements arranged in a tree layout.
- JavaScript can access this tree and manipulate elements

Element Node

Text

Attributes



Document Object Model (DOM) (RECAP)

Java**S**cript is…

… a core technology of the World Wide Web

… a scripting language, interpreted by the browser

… weakly typed

… prototype based

2 ways for embedding JavaScript in HTML:

- Use the HTML-tag <script> in the head or in the body to write your code.

```
<script> console.log("Hello World") </script>
```

- External file (e.g., script.js)

```
<script src="script.js"></script>
```

Access elements:

- By ID:

```
document.getElementById("uniqueID");
```

- By Class Name:

```
document.getElementsByClassName("classID");
```

- By Tag:

```
document.getElementsByTagName("tag-name");
```

Access elements:

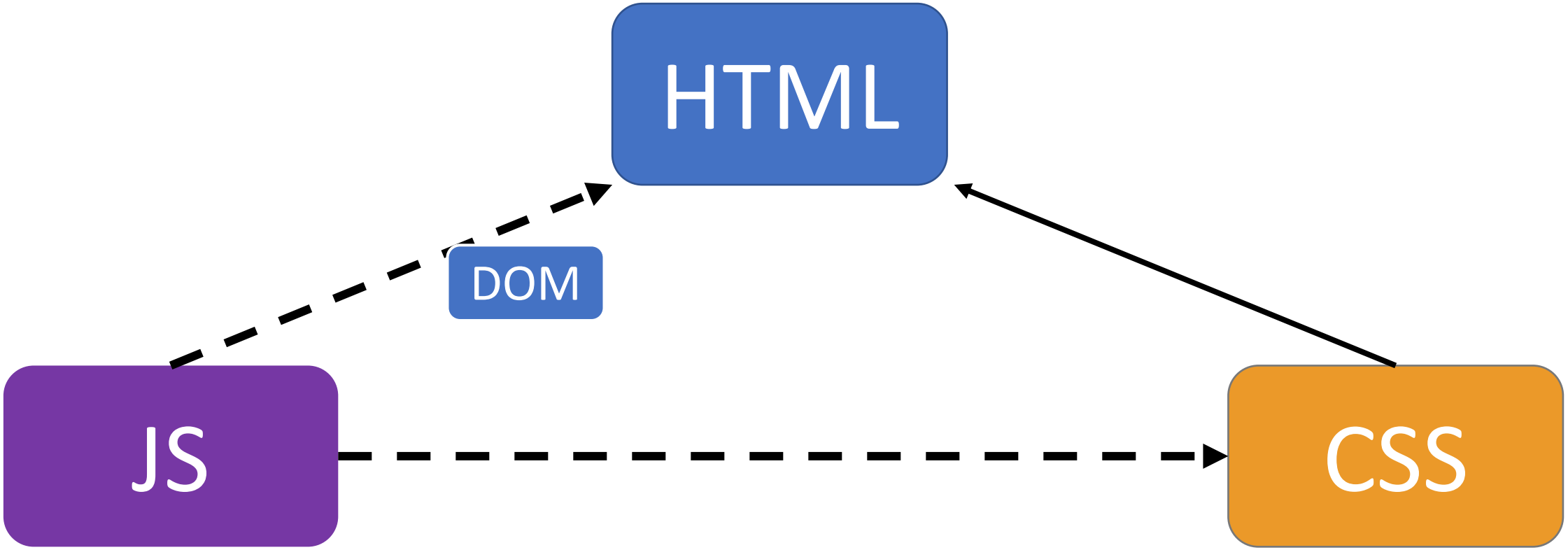
```
var header = document.getElementById("header");
header.setAttribute("style", "color:red");

var tds = document.getElementsByTagName("td");
tds[1].setAttribute("style", "color:red");
```

Example:

```
var newElement = document.createElement("div");
newElement.setAttribute("id", "newElement");
document.getElementsByTagName("body")[0].appendChild(new
Element);
```

JavaScript: DOM Manipulation Example (RECAP)



Summary (RECAP)

A simple example of a **scatterplot** can be found here:

`seminar/code/code-D3/index-1_PlainJS.html`

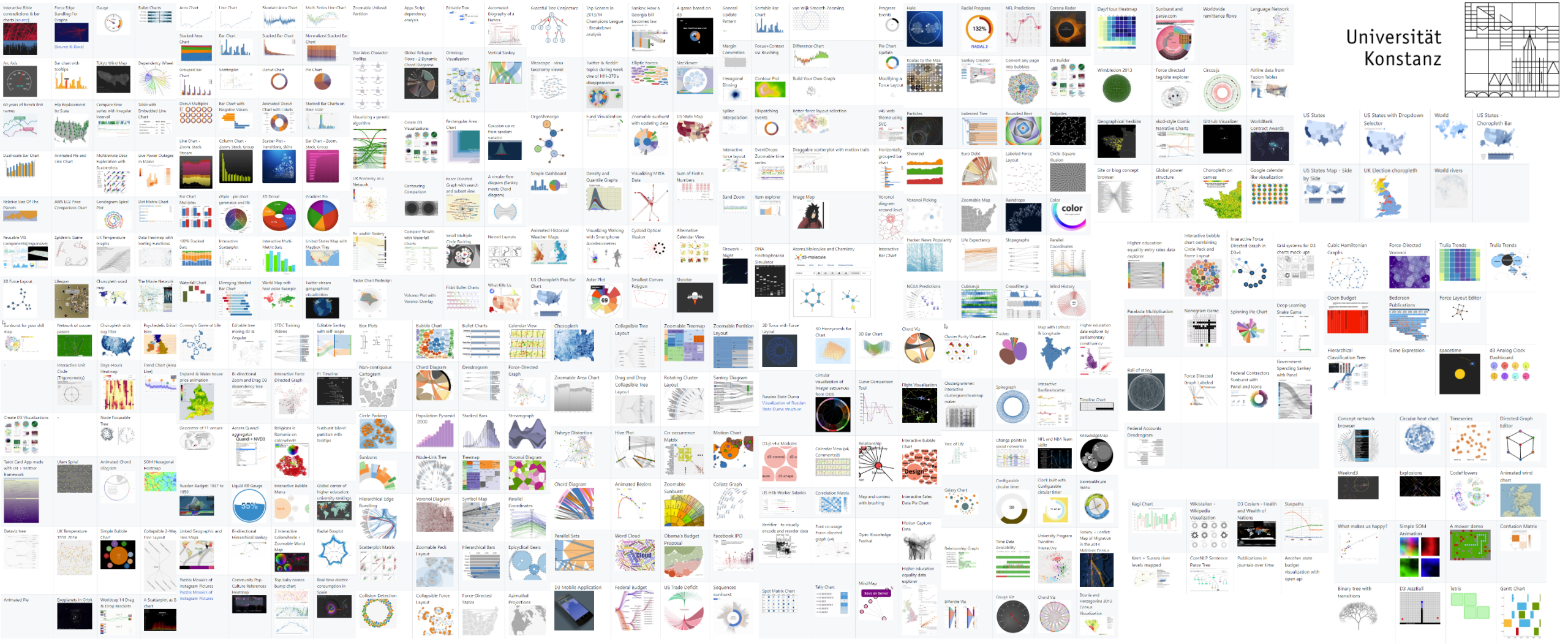
https://www.w3schools.com/

References and Examples for HTML, CSS, JavaScript

https://jsfiddle.net/

An online playground for your JavaScript, HTML, CSS

Summary



https://github.com/d3/d3/wiki/gallery

D3 - Gallery

1. Visit https://d3js.org
2. Download the latest Version
3. Copy the file **d3.js** into your web-project
4. Link d3.js in your **index.html** using the <script> tag

```
<script src="lib/d3.js"></script>
```

Hint: Be sure to specify the path relative to the **index.html** file

Documentation: https://github.com/d3/d3/blob/master/API.md

D3 - Installation

`d3.select` / `d3.selectAll` allows the selection of DOM elements.

Examples:

1. Select by HTML-Tag: `d3.select("p")`
2. Select by class-name: `d3.selectAll(".paragraphs")`
3. Select by ID: `d3.select("#paragraph")`

- `d3.select` selects the first element in the document.
- `d3.selectAll` selects all elements matching your selector.
- If there is no matching element, `d3.select/d3.selectAll` returns an empty selection.

`selection.attr` allows to change HTML attributes of the selection.

`selection.style` similarly allows to change the style of the selection via inline CSS.

```
var body = d3.select("body");      // Retrieve body DOM-node
var p = body.append("p");          // Append paragraph to body
p.text("New paragraph!");          // Set text of paragraph
p.attr("id", "my-paragraph");      // Set the HTML attribute id
p.style("color", "red");           // Set the color via inline css
```

D3 – Interaction with the DOM - Edit Attributes/Styles

```
var body = d3.select("body");      // Retrieve body DOM-node
var p = body.append("p");          // Append paragraph to body
p.text("New paragraph!");          // Set text of paragraph
```

Is equivalent to:

```
d3.select("body")                  // Retrieve body DOM-node
  .append("p")                     // Append paragraph to body
  .text("New paragraph!");         // Set text of paragraph
```

Chain syntax can significantly reduce the lines of code

D3 – Paradigms – Method Chaining

`selection.data(dataset)` joins the specified array of *dataset* with the selected elements of the *selection* and returns the so-called *update* selection.

The specified *dataset* is an array of arbitrary values (*e.g.*, numbers or objects), or a function that returns an array of values for each group. When data is assigned to an element, it is stored in the property `__data__`.

D3 – Paradigms – Data Binding - data()

selection.enter() returns the enter selection: placeholder nodes for each datum that had no corresponding DOM element in the selection.

- The enter selection is empty for selections not returned by *selection*.data(*dataset*)
- The enter selection is typically used to create "missing" elements corresponding to new data. For example, to create <p> elements from an array of numbers.

```
var dataset = [5, 10, 15, 20, 25];

d3.select("body").selectAll("p").data(dataset).enter()
  .append("p").text("New paragraph!");
```

```
> console.log(d3.selectAll("p"))
[▼Array[5]                    ]
   ▶ 0: HTMLParagraphElement
   ▶ 1: HTMLParagraphElement
   ▶ 2: HTMLParagraphElement
   ▶ 3: HTMLParagraphElement
   ▶ 4: HTMLParagraphElement
     length: 5
   ▶ parentNode: HTMLDocument
   ▶ __proto__: Array[0]
```

Reference to javascript DOM representation

```
> console.log(d3.selectAll("p"))
[▼Array[5]
   ▼ 0: HTMLParagraphElement
       __data__: 5
       align: ""
     ▶ attributes: NamedNodeMap
```

When data is assigned to an element, it is stored in the property *__data__*

D3 – Paradigms – Data Binding - Example

```
var dataset = [5, 10, 15, 20, 25];

d3.select("body").selectAll("p").data(dataset).enter()
  .append("p").text("New paragraph!")
  .text(function(d) {
    return "I can count to " + d;
  });
```

- Specifying an anonymous function as the parameter for the text function allows access to data values.
- Anonymous functions can be used in many cases and for many d3 functions such as *text(), attr(), style(), etc.*

D3 – Paradigms – Anonymous Functions as Parameters

A simple example of **data binding** can be found here:

`seminar/code/code-D3/index-2_DataBinding.html`

A simple example of **scaling** can be found here:

`seminar/code/code-D3/index-3_Scales.html`

`d3.dsv` allows us to load data in a format of delimeter-separated values (e.g. csv files). Further function can load: images, html, json, text, xml

Example:

```
d3.dsv(",", "test.csv", function(d) {
  // This function is called for every data record (row)
  return {
    length: +d.Length // convert "Length" column to number
  };
}).then(function(data) {
  // This function is called once the data is loaded
});
```

See: https://github.com/d3/d3-fetch

- JavaScript (the Browser) does not allow to load files from your local file-system (fs) for security reasons.

- How can we then load data in other formats? (e.g., json/csv)

  A webserver allows you to access your fs, such that you can load data from your local fs and transmit it to your web-application via HTTP.

Webserver to Load Data

1. Visit https://python.org
2. Download latest version of Python 3.7 for your platform
3. Follow the installation steps
4. Open a console/PowerShell

   Make sure python is installed:

   > `python --version`

   Should output the installed version (e.g., Python 3.6.1)

Python-Webserver Installation

1. Open a Console/PowerShell
2. Navigate to the root folder of your web project
3. Start the python webserver

```
> python -m http.server    OR    python3 -m http.server
    Output: Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/)
...
```

4. Check that the webserver is running by opening a browser and visit http://localhost:8000 – You should see the files and folders of the directory you started the webserver in.

Python-Webserver Start

Given the file

`seminar/code/code-D3/index-4_LoadCSV.html`

implement a simple **scatterplot**.

Think about:

1. webserver
2. data binding
3. SVG elements
4. scales

- API Reference: https://github.com/d3/d3/blob/master/API.md
- Gallery: https://github.com/d3/d3/wiki/Gallery
- Examples: https://bl.ocks.org/mbostock
- Tutorials: https://github.com/d3/d3/wiki/Tutorials

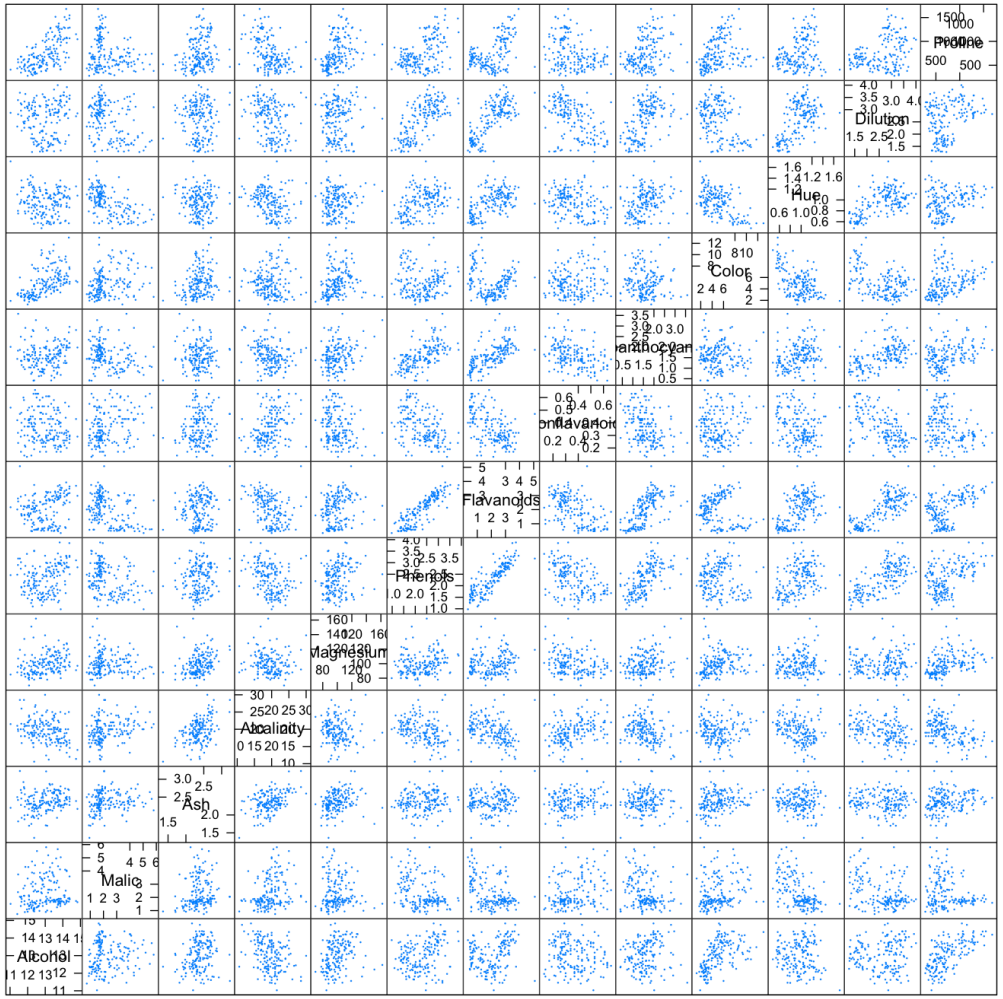
Due to the large community using d3 you will find d3 also on stackoverflow (31,684 questions tagged), observablehq, google, youtube, etc.

Resources

Check out 

https://www.knime.com/downloads/download-knime

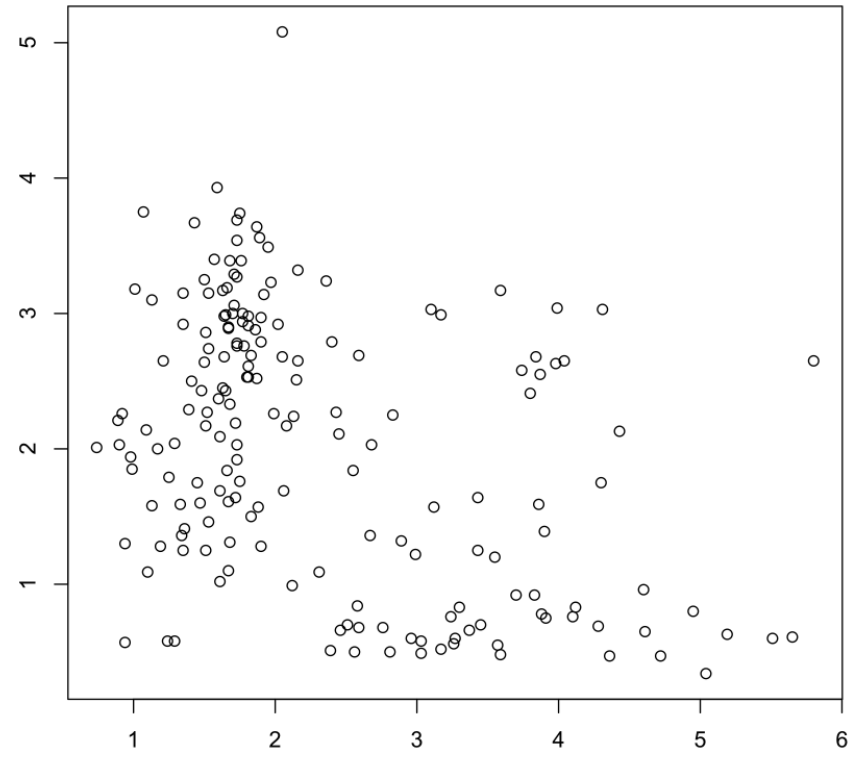
You do not like JavaScript...

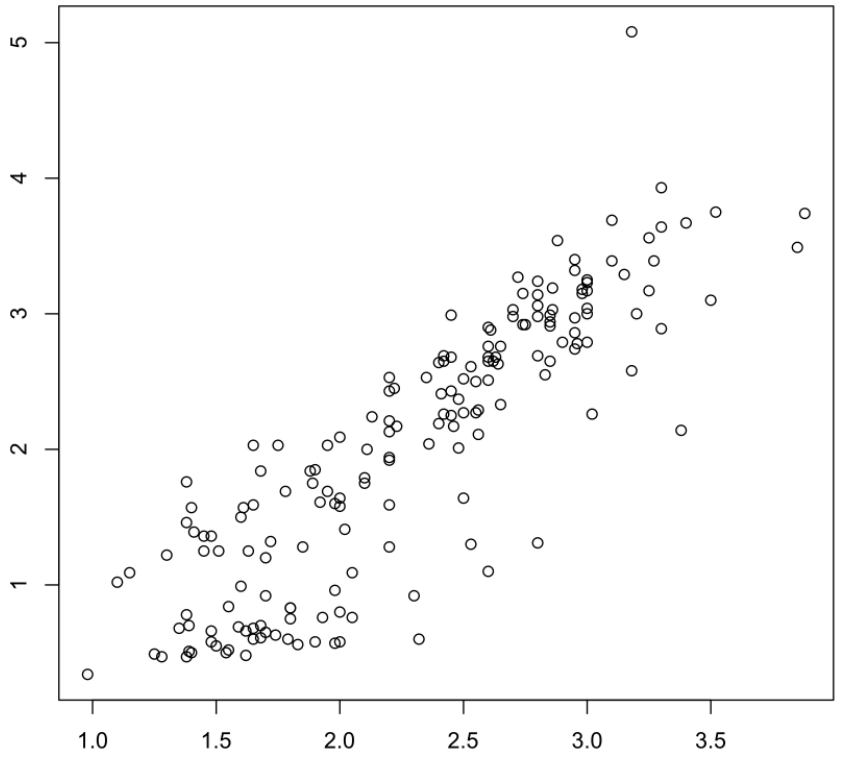


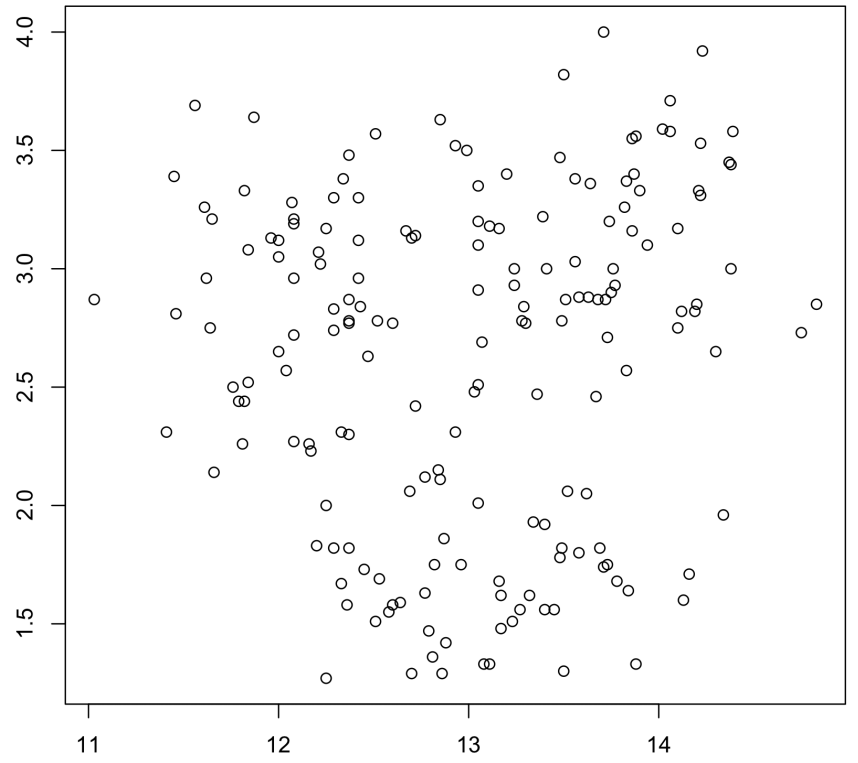
https://observablehq.com/@d3/scatterplot-matrix

Scatterplot Matrix

How to decide, which scatterplot is best?







Scatterplot Matrix

- Idea: reduce the size of scatterplot matrices.
  - Show only interesting plots

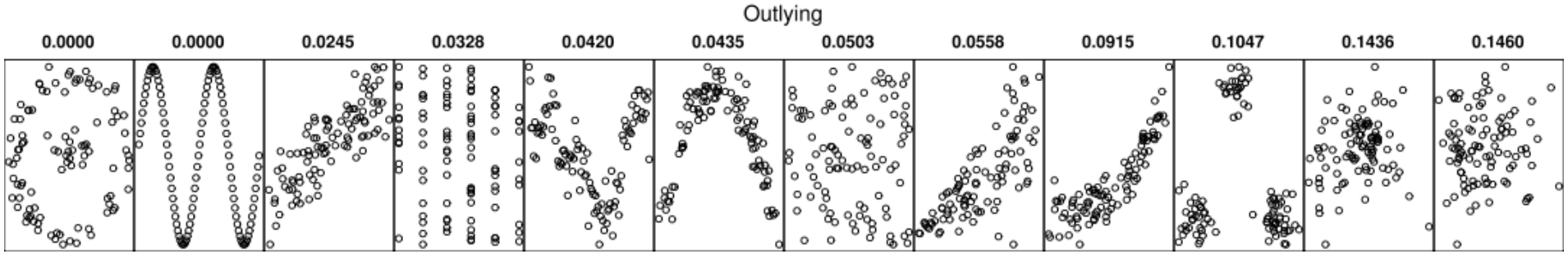
- Measurements to describe properties of scatterplots
  - Outlier
  - Density
  - Shape
  - Relation

Scatterplot Matrix - Scagnostics

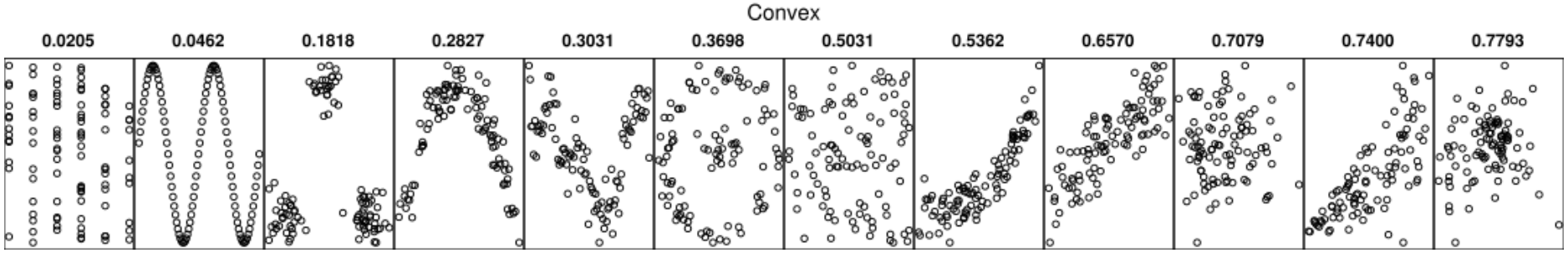
Tukey and Tukey, 1988

Use the minimal spanning tree (MST)

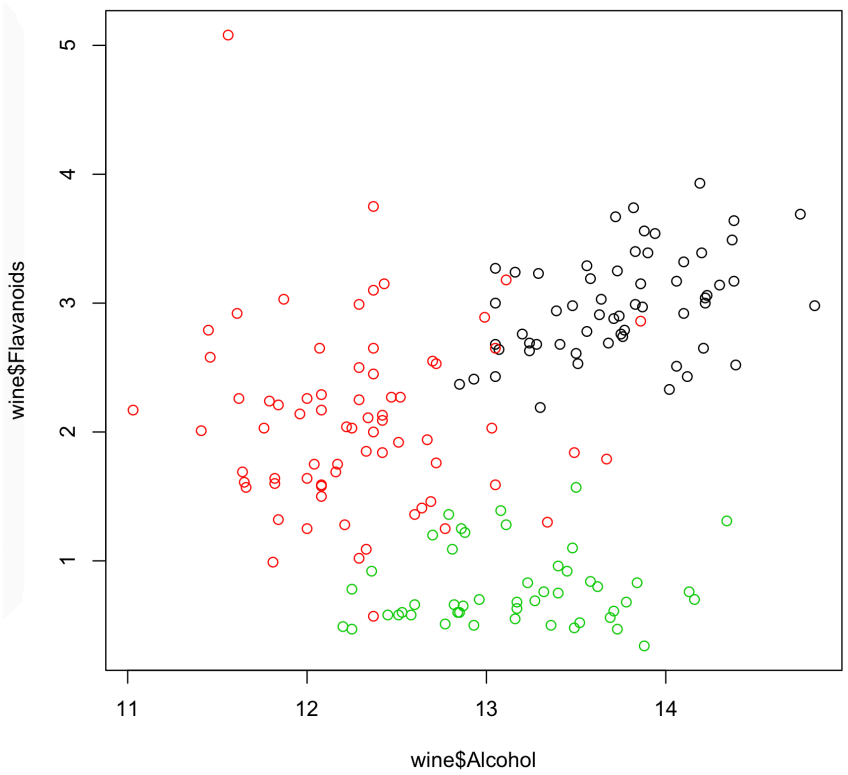
- Portion of long edges in the MST

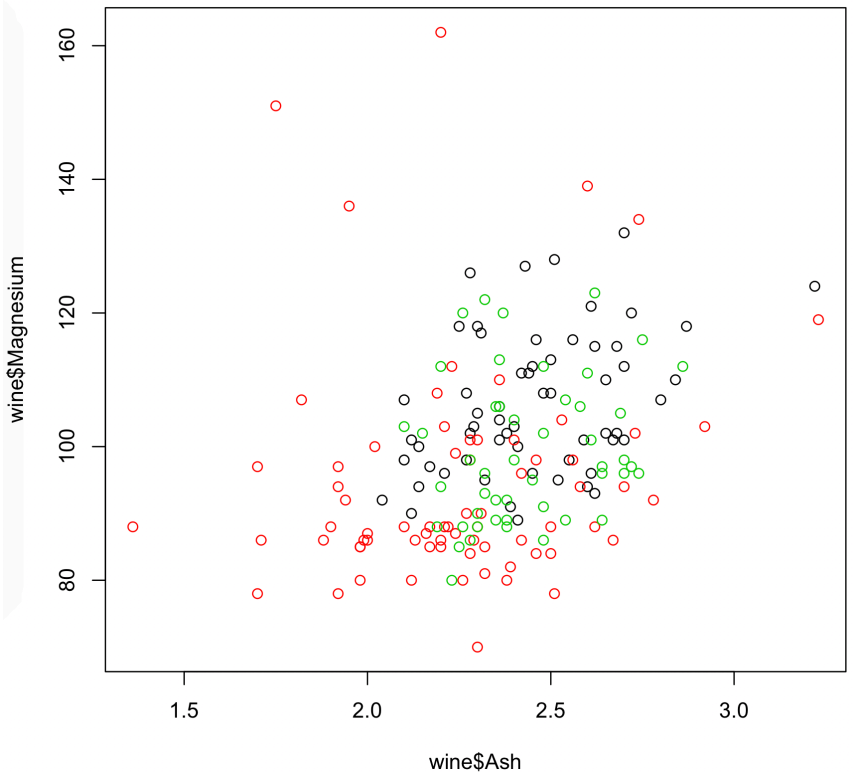


- Ratio between alpha shape and convex hull



A view is consistent when the classes are mapped to regions that are visually separable.



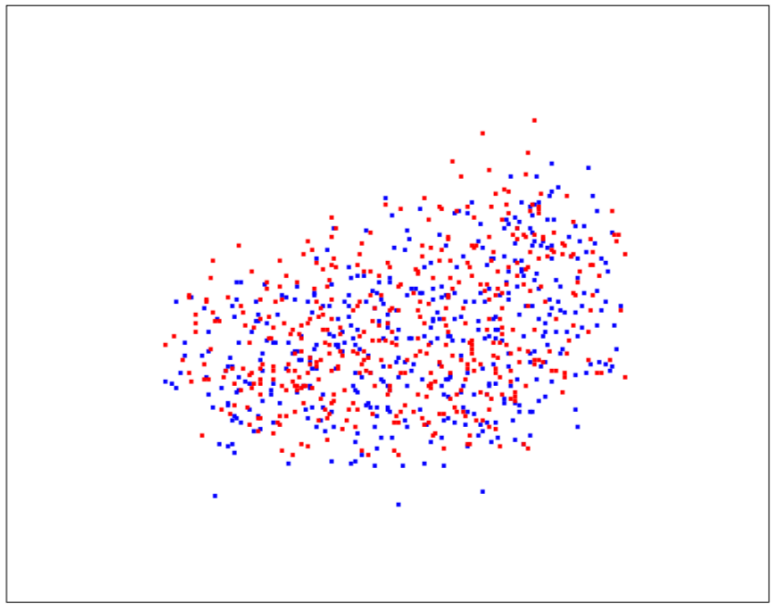


Scatterplot Matrix - Class Consistency

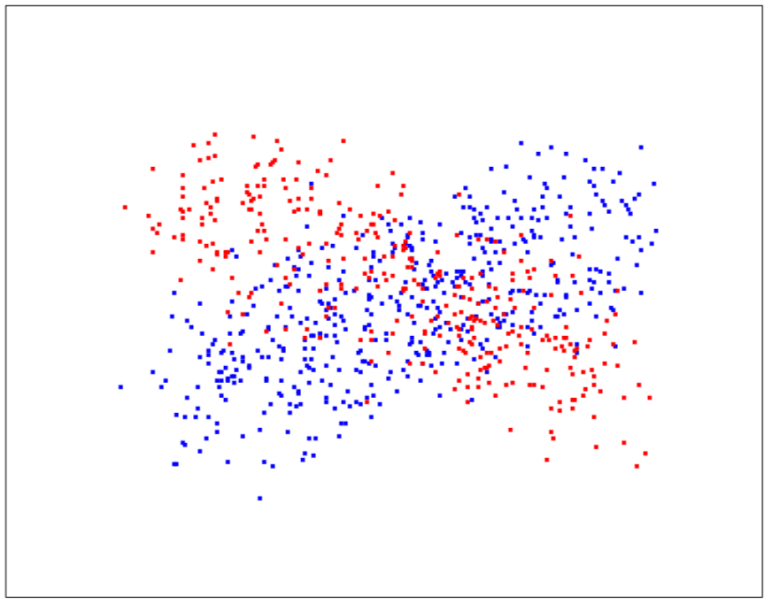
How can we compute the class consistency?

- Use a clustering algorithm like e.g., k-means
- Compute centroid distance (CD) in the projected space.
- **Distance consistency** is the classification error of class members using CD
- **Distribution consistency** is a weighted measure (according to the amount of data) based on the entropy in a region

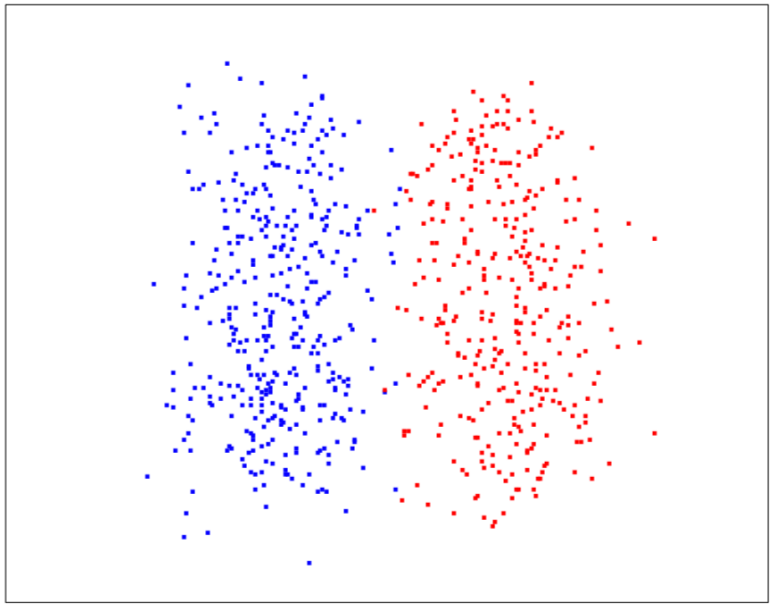
Distribution consistency



(e) 15

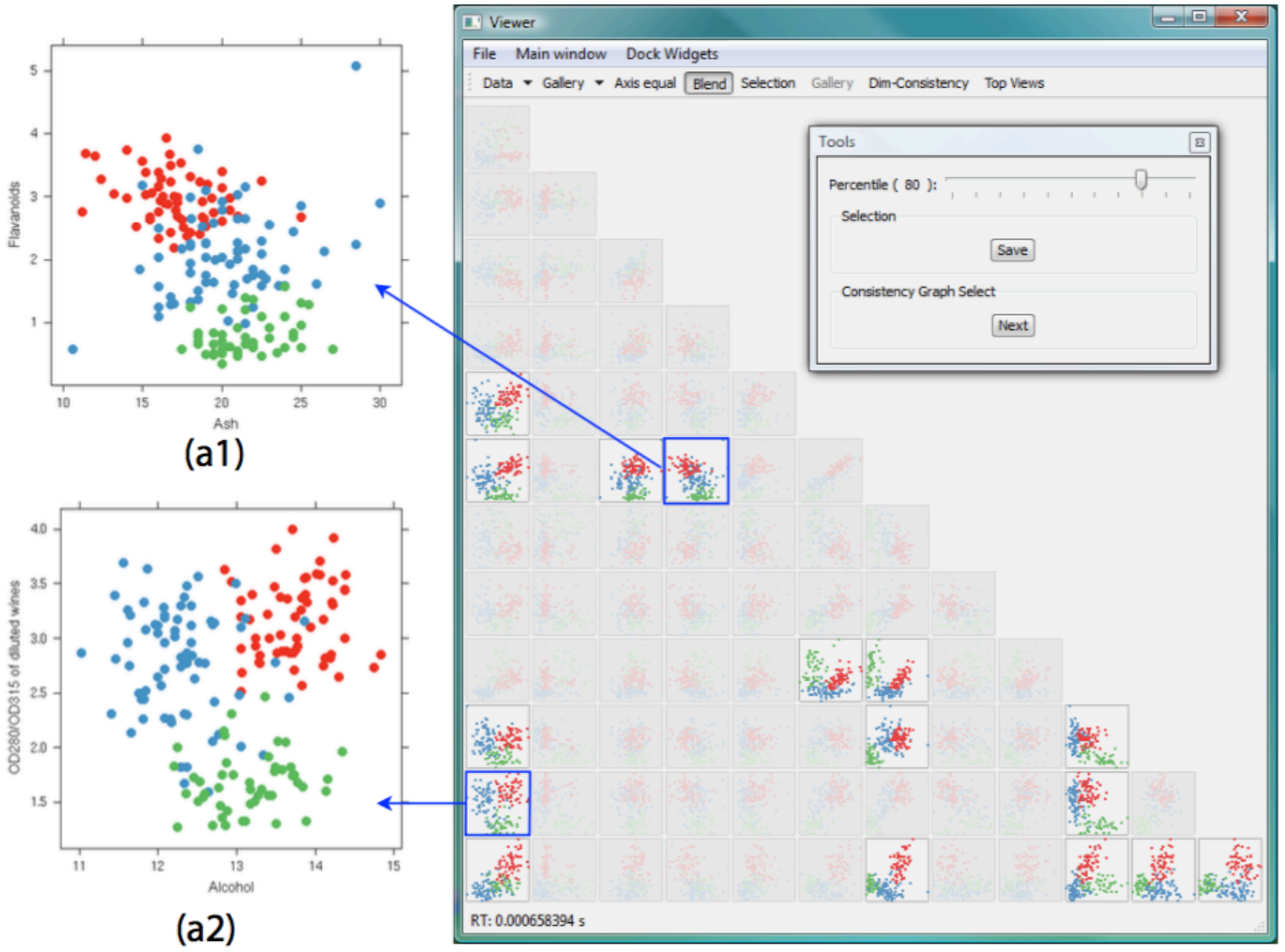


(c) 51



(a) 99

Sips et al., 2009

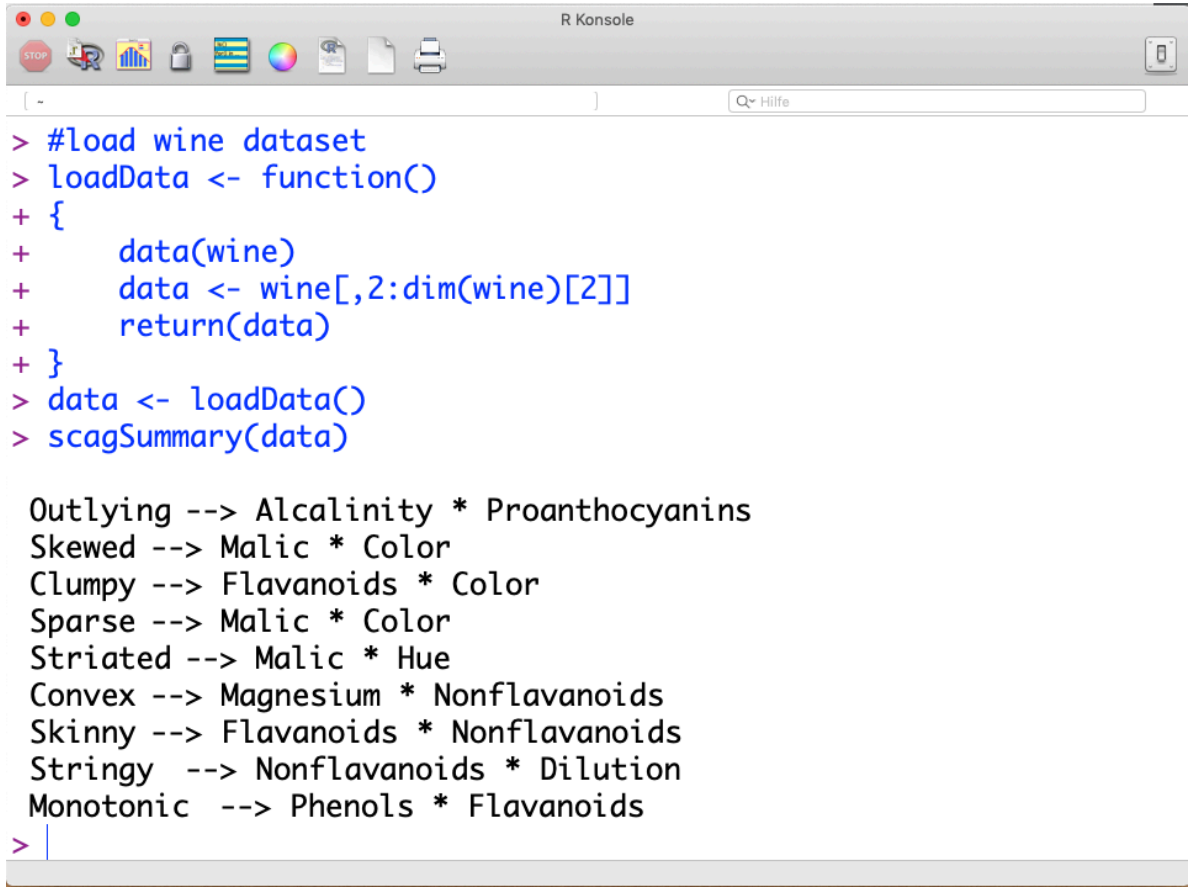


(a) Wine (13 Dimensions)

Scatterplot Matrix - Class Consistency

R-Code: 04a_Scagnostics.r

R-Code: 04b_DistributionConsistency.r



- Use some metrics (scagnostics,…) to help you focus on interesting aspects of your scatterplot matrix.
- Class consistency can be used to help you interpret labeled data.
- In case no labels are available perform a cluster analysis beforehand.

Guidelines – Visualization Foundation

Tukey and Tukey, 1988
Computer graphics and exploratory data analysis: An introduction
JW Tukey, PA Tukey - The Collected Works of John W. Tukey: Graphics: 1965-1985, Wadsworth Publishing Company, 1988

Ward and Yang, 2004
Interaction Spaces in Data and Information Visualization
Matthew O. Ward and Jing Yang: Eurographics Symposium on Visualization: 137-145, Eurographics Association, 2004

Sips et al., 2009
Selecting good views of high-dimensional data using class consistency
Mike Sips & Boris Neubert & John P. Lewis & Pat Hanrahan: Computer Graphics Forum: 831-838, Wiley Online Library, 2009

Mayorga and Gleicher, 2013
Splatterplots: Overcoming Overdraw in Scatter Plots
A. Mayorga, M Gleicher - Transaction on Visualization and Computer Graphics, IEEE, 2013

Skau and Kosara, 2016
Arcs, angles, or areas: Individual data encodings in pie and donut charts
Skau, Drew and Kosara, Robert - Computer Graphics Forum, 2016

Sarikaya and Gleicher, 2018
Scatterplots: Tasks, Data, and Designs
A Sarikaya, M Gleicher - Transaction on Visualization and Computer Graphics, IEEE, 2018

References