Classification of Visualization Systems

Univariate analysis

- provides summary statistics for each field in the raw data set (or) summary only on one variable.
- Ex:- CDF,PDF,Box plot, Violin plot.

Bivariate analysis

- performed to find the relationship between each variable in the dataset and the target variable of interest (or) using 2 variables and finding the relationship between them.
- Ex:-Box plot, Violin plot.

Multivariate analysis

- performed to understand interactions between different fields in the dataset (or) finding interactions between variables more than 2.
- Ex:- Pair plot and 3D scatter plot.

Example

- About Dataset:
  - The dataset contains cases from a study that was conducted between 1958 and 1970 at the University of Chicago's Billings Hospital on the survival of patients who had undergone surgery for breast cancer.
- Attribute Information:
  - Age of patient at time of operation (numerical)
  - Patient's year of operation (year - 1900, numerical)
  - Number of positive axillary nodes detected (numerical). Lymph node in the area of the armpit (axilla) to which cancer has spread.
  - Survival status (class attribute):
    - 1 = the patient survived 5 years or longer
    - 2 = the patient died within 5 year

Time Series

What is a Time Series Data?

- Anything that is observed or measured at many points in time forms a time series.
- Many time series are fixed frequency, where data points occur at regular intervals according to some rule, such as every 15 seconds, every 5 minutes, or once per month.
- Time series can also be irregular without a fixed unit of time or offset between units.

How you mark?

- How you mark and refer to time series data depends on the application
- One of the following:
  - Timestamps, specific instants in time
  - Fixed periods, such as the month January 2007 or the full year 2010
  - Intervals of time, indicated by a start and end timestamp. Periods can be thought of as special cases of intervals
  - Experiment or elapsed time; each timestamp is a measure of time relative to a particular start time (e.g., the diameter of a cookie baking each second since being placed in the oven)

Text Data

Ref: https://towardsdatascience.com/a-complete-exploratory-data-analysis-and-visualization-for-text-data-29fb1b96fb6a

Visualization Techniques for Trees, Graphs, and Networks

Reference: Storytelling with Data_ A Data Visualization Guide for Business Professionals

Displaying Hierarchical Structures

- We can divide these techniques into two classes of algorithms:
- space-filling
- non-space-filling

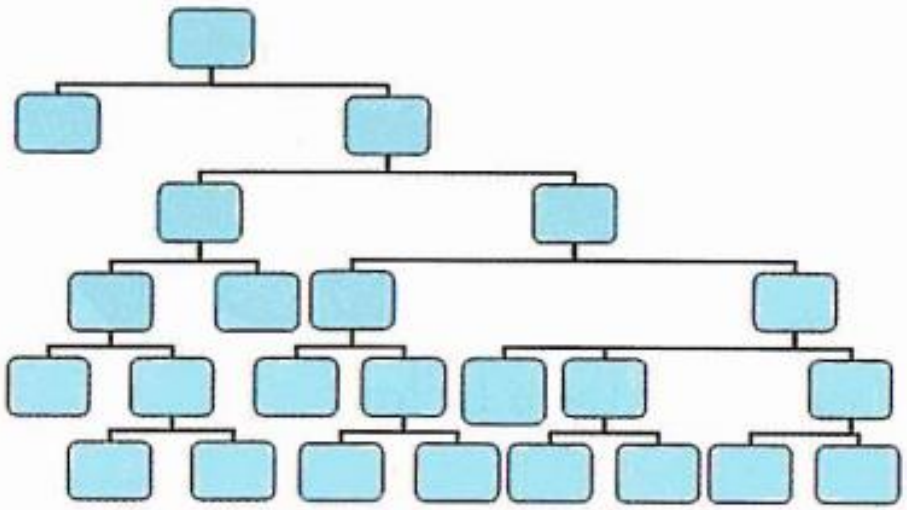
space-filling methods

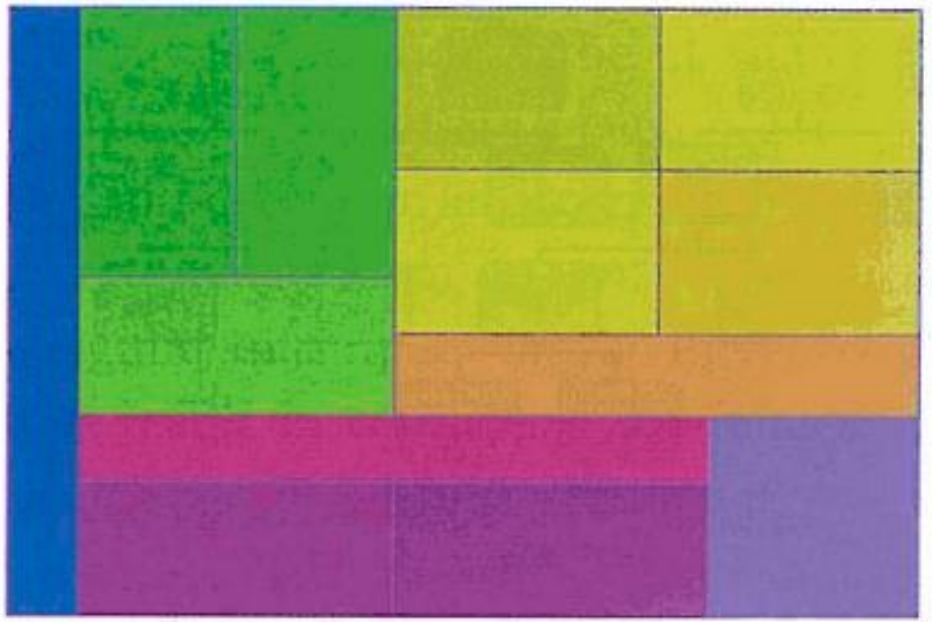
- space-filling techniques make maximal use of the display space.
- This is accomplished by using juxtapositioning* to imply relations
- The two most common approaches to generating space-filling
- hierarchies are rectangular and radial layouts.

*the act or an instance of placing two or more things side by side often to compare or contrast or to create an interesting effect

Treemaps

- Treemaps and their many variants are the most popular form of rectangular space-filling layout.
- In the basic treemap, a rectangle is recursively divided into slices, alternating horizontal and vertical slicing, based on the populations of the subtrees at a given level.





Pseudo Code

```
Start: Main Program
  Width = width of rectangle
  Height = height of rectangle
  Node = root node of the tree
  Origin = position of rectangle, e.g., [0,0]
  Orientation = direction of cuts, alternating between horizontal and vertical
  Treemap(Node, Orientation, Origin, Width, Height)
End: Main Program

Treemap(node n, orientation o, position orig, hsize w, vsize h)
  if n is a terminal node (i.e., it has no children)
     draw-rectangle(orig, w, h)
     return
  for each child of n (child_i), get number of terminal nodes in subtree
  sum up number of terminal nodes
  compute percentage of terminal nodes in n from each subtree (percent-i)
  if orientation is horizontal
     for each subtree
        compute offset of origin based on origin and width (offset-i)
        treemap(child_i, vertical, orig + offset-i, w * percent-i, h)
  else
     for each subtree
        compute offset of origin based on origin and height (offset-i)
        treemap(child_i, horizontal, orig + offset-i, w, h * percent-i)
End: Treemap
```

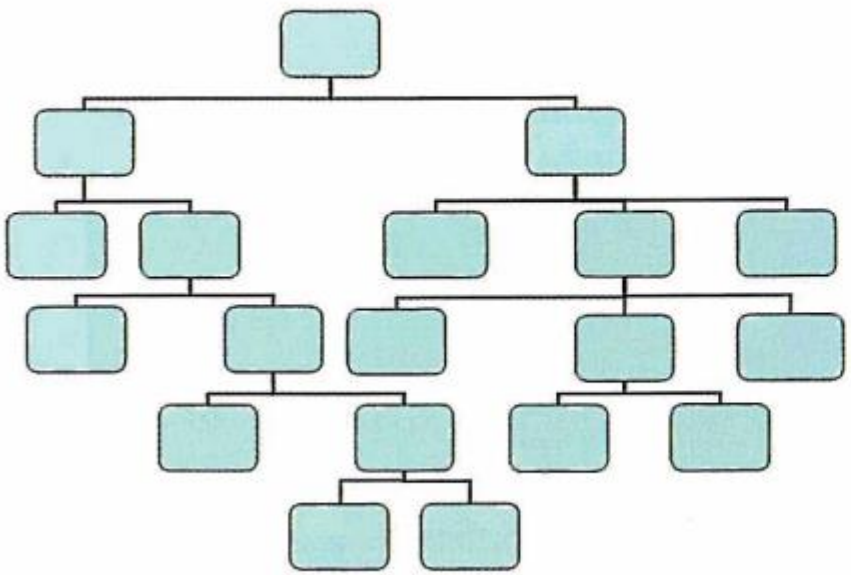
Radial space-filling

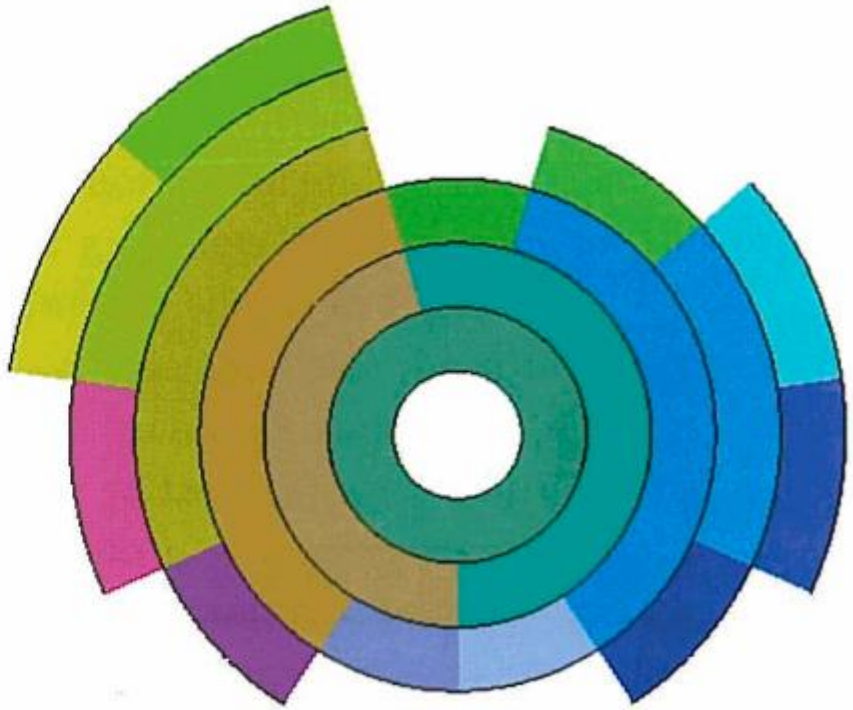
- Radial space-filling hierarchy visualizations, sometimes referred to as sunburst displays
- Root of the hierarchy in the center of the display and use nested
- rings to convey the layers of the hierarchy.
- Each ring is divided based on the number of nodes at that level.
- These techniques follow a similar strategy to treemaps, in that the number of terminal nodes in a subtree determines the amount of screen space that will be allocated for it.

Pseudo Code

```
Start: Main Program
  Start = start angle for a node (initially 0)
  End = end angle for a node (initially 360)
  Origin = position of center of sunburst, e.g., [0,0]
  Level = current level of hierarchy (initially 0)
  Width = thickness of each radial band - based on max depth and display size
  Sunburst(Node, Start, End, Level)
End: Main Program

Sunburst(node n, angle st, angle en, level l)
  if n is a terminal node (i.e., it has no children)
     draw_radial_section(Origin, st, en, l * Width, (l+1) * Width)
     return
  for each child of n (child-i), get number of terminal nodes in subtree
  sum up number of terminal nodes
  compute percentage of terminal nodes in n from each subtree (percent_i)
  for each subtree
     compute start/end angle based on size of subtrees, order, and angle range
     Sunburst(child-i, st_i, en_i, l+1)
```





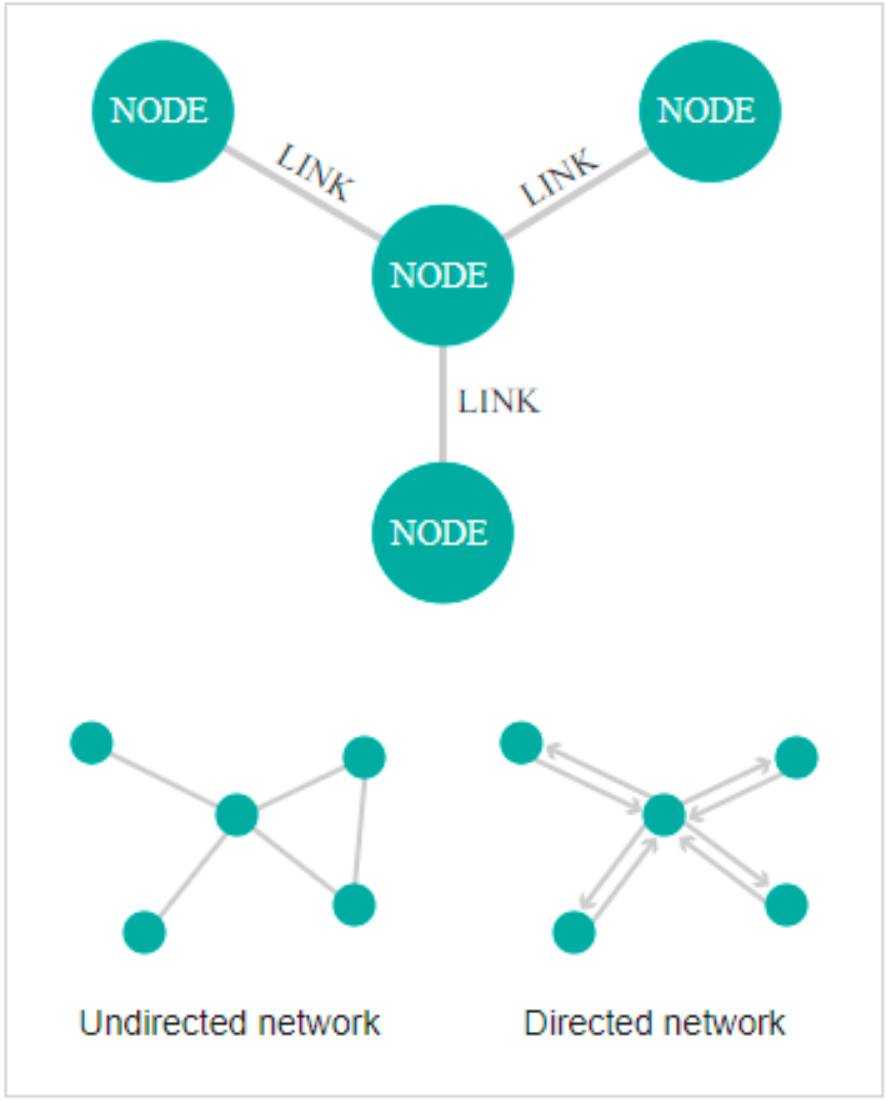
Non-Space-Filling Methods

- The most common representation used to visualize tree or hierarchical relationships is a node-link diagram.
- Organizational charts, family trees, and tournament pairings are just some of the common applications for such diagrams.

- Example: https://medium.com/@ahsenparwez/building-a-family-tree-with-python-and-graphviz-e4afb8367316

Displaying Arbitrary Graphs/Networks

- **Node-Link Graphs**
- This type of visualization shows how things are interconnected through the use of nodes / vertices and link lines to represent their connections and help illuminate the type of relationships between a group of entities.
- Typically, nodes are drawn as little dots or circles, but icons can also be used. Links are usually displayed as simple lines connected between the nodes.



- PC: https://datavizcatalogue.com/methods/images/anatomy/SVG/network_diagram.svg

Visualizing Vector Fields

- Vector fields contain vector information for every point in space. For example, air flow data inside a wind tunnel is a vector field.
  - Velocity Plot (Quiver Plot)
  - Velocity Plot (Cone Plot)
  - Streamlines
  - Streamslice
  - Stream Tube and Stream Ribbon