Assignment1

COMP 5048

ZWAN0168

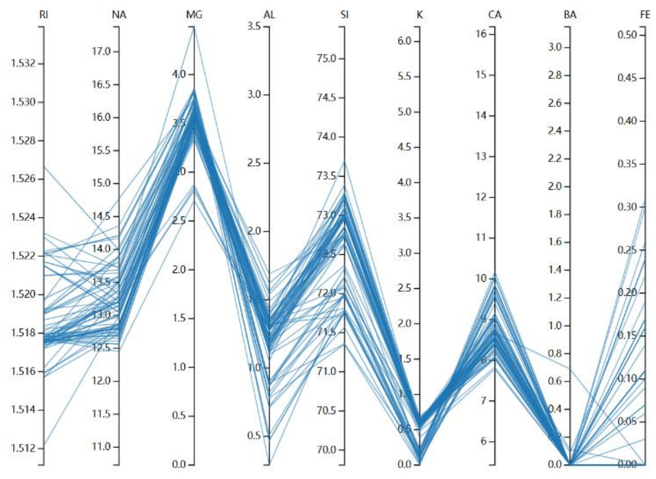
550116286

Assignment 1
ZWAN0168
ZHENYU WANG 550116286

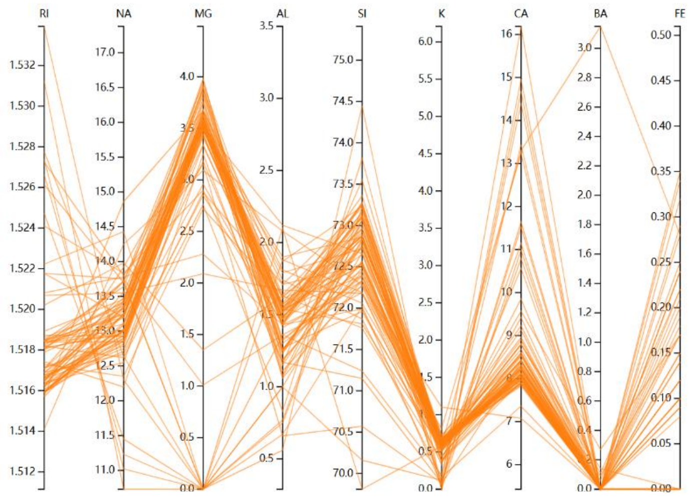
Content：

Assignment 1
ZWAN0168
ZHENYU WANG 550116286

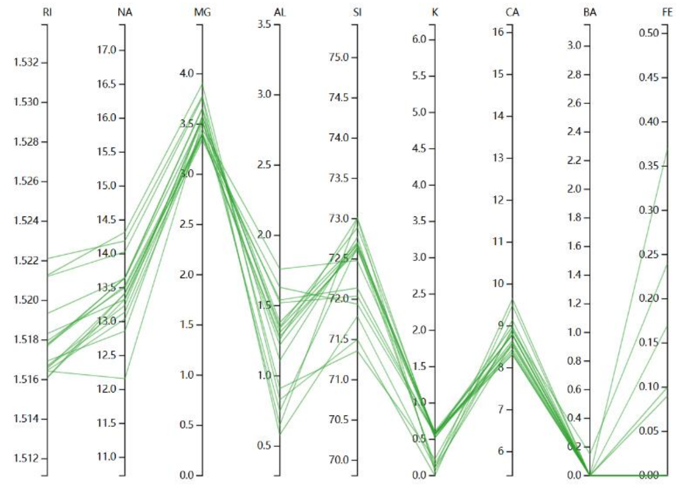
Visual Analysis 1(parallel coordinates)



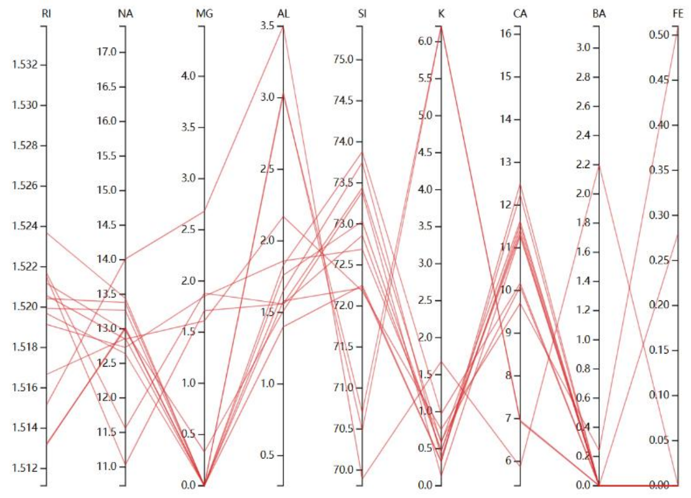
Type 1



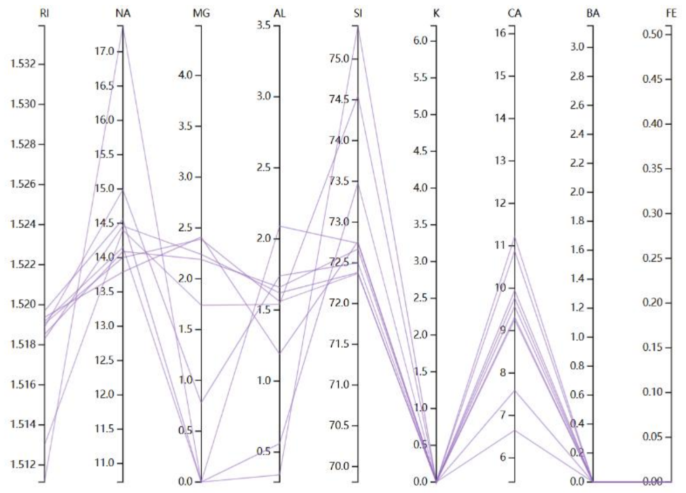
Type 2



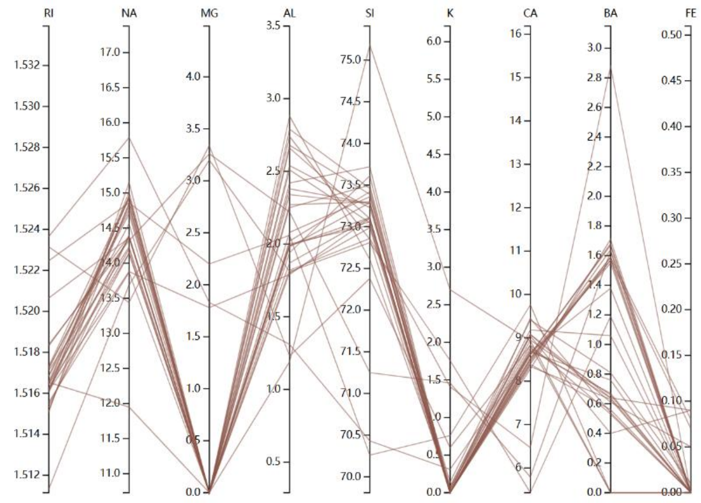
Type3



Type5

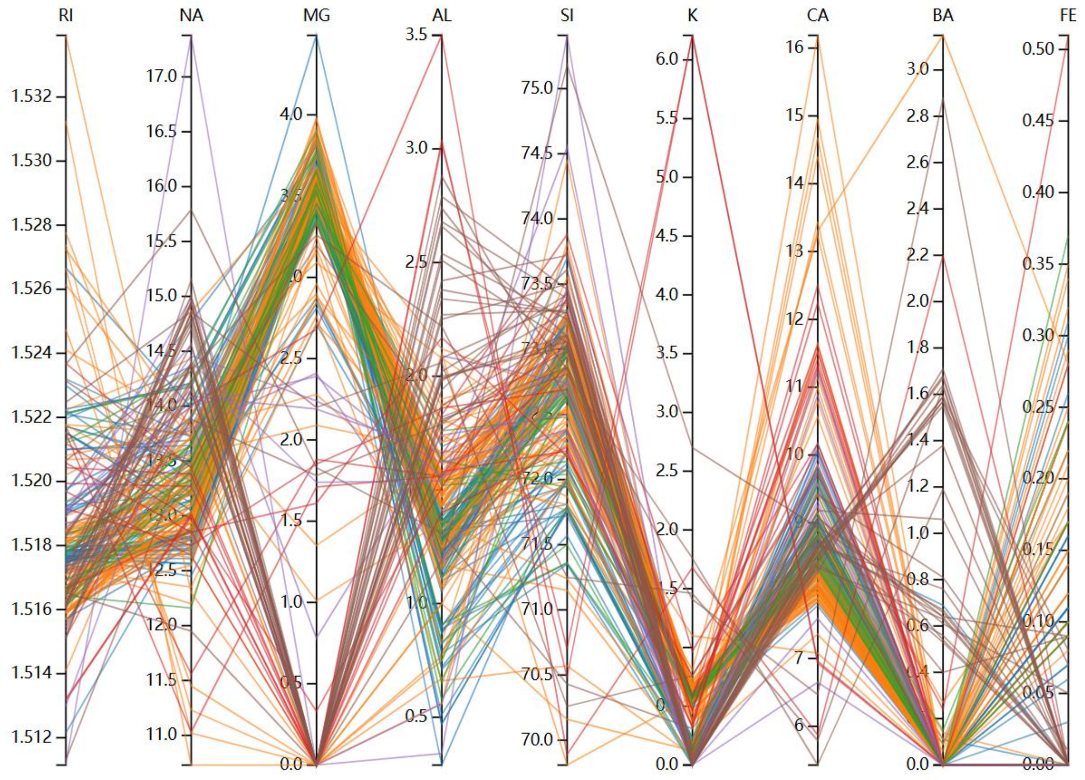


Type6



Type 7

Assignment 1
ZWAN0168
ZHENYU WANG 550116286



Discussion on Visual Analysis 1

Design:

The analysis implemented an interactive parallel coordinates visualization using D3.js to process glass composition data containing nine elements: refractive index (RI), sodium (Na), magnesium (Mg), aluminum (Al), silicon (Si), potassium (K), calcium (Ca), barium (Ba), and iron (Fe). Six glass types were differentiated through color-coded polylines connecting corresponding element values across vertical axes. The system incorporated axis scaling to accommodate diverse measurement units (RI values vs. oxide percentages), interactive range selection filters via rectangular brushing, and z-score normalization for oxide concentration comparisons. Path simplification algorithms reduced visual overlap between lines while preserving elemental relationship patterns, particularly between RI and metallic components (Na, Mg, Ba) across all six glass categories.

Evaluation:

The parallel coordinates visualization demonstrates distinct compositional variations among six glass types, differentiated by colored lines. Analysis reveals significant differences in elemental composition: certain types exhibit elevated magnesium (Mg) content compared to others, while others show higher sodium (Na) and aluminum (Al) concentrations. A consistent pattern emerges across all glass types, with silicon (Si) constituting the predominant component. As noted by Shakhgildyan et al. (2020), "The most common class of glasses is oxide glasses... most glasses are produced on their basis [of silicon oxides]."

Key differentiating elements include potassium (K), calcium (Ca), and barium (Ba), which show categorical specificity, though overlapping patterns in refractive index (RI) and iron (Fe) content create classification ambiguities. A subset of specimens display atypical elemental ratios in heavy metal components, deviating from their categorical norms.

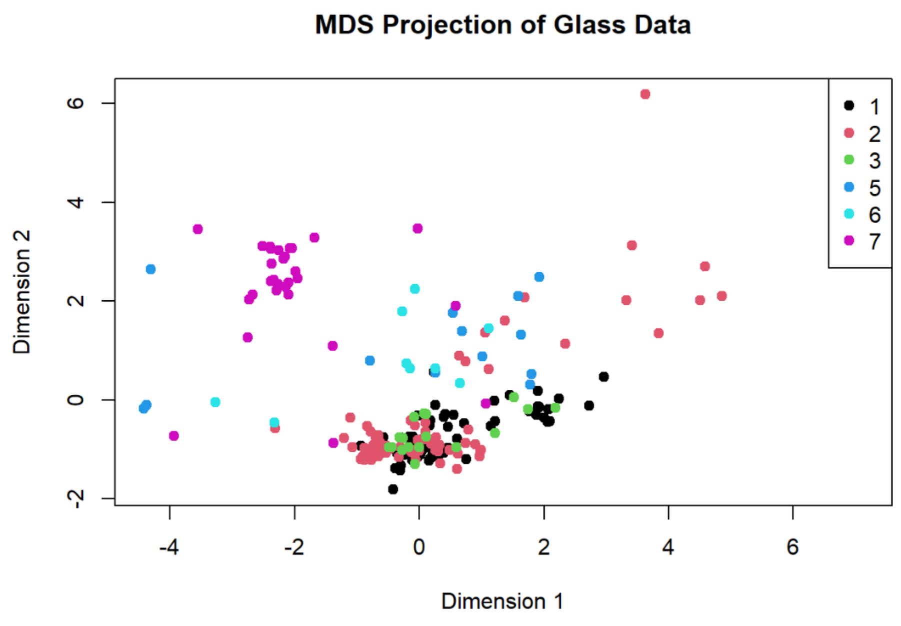
Pros:

The visualization provides flexibility to filter data through interactive brushing, enabling clear presentation of multivariate relationships through color differentiation. Features and anomalies can be effectively observed via line pattern variations.

Cons:

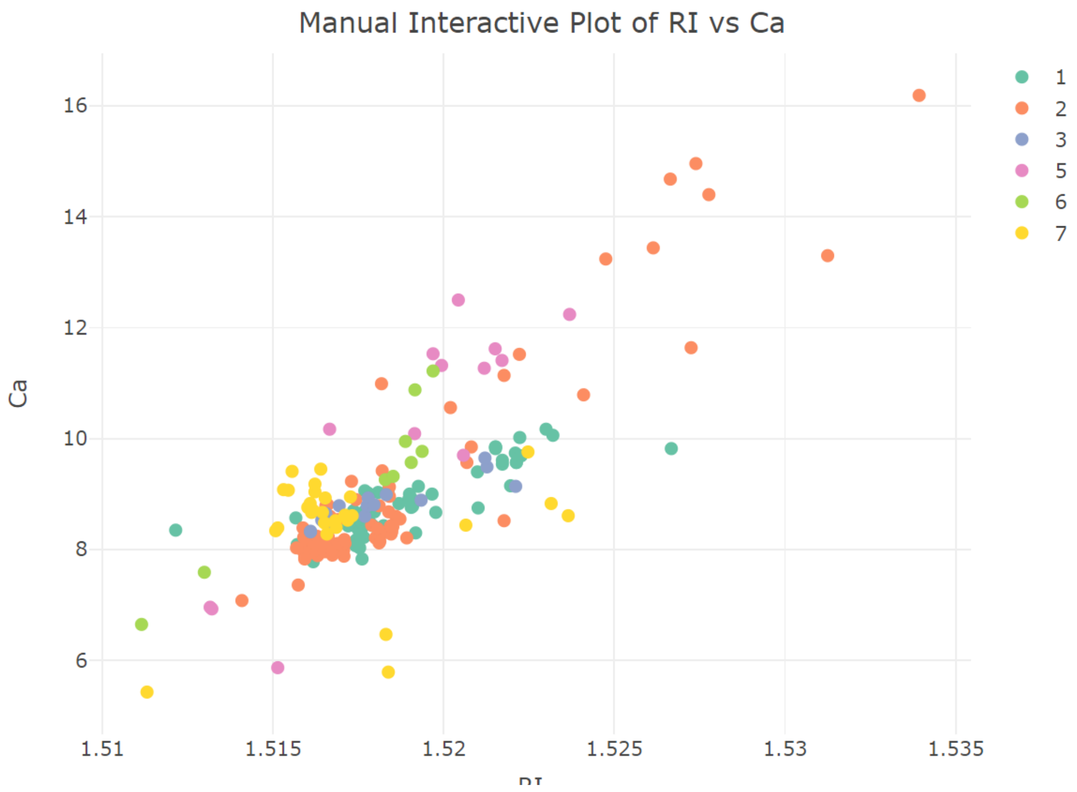
Large datasets reduce interpretability due to line overlap, obscuring individual data details. Static renderings lose interactive capabilities, preventing precise analysis of overlapping elements.

Visual Analysis 2 (MDS)



```
##                RI           Na           Mg           Al           Si            K           Ca
## Dim1 0.8639224 -0.4090426  0.1757092 -0.6793596 -0.3626290 -0.3475869 0.7801403
## Dim2 0.4090433  0.3870892 -0.8498611  0.4226860 -0.2220718 -0.2204556 0.4945173
##                Ba           Fe
## Dim1 -0.3967607  0.2944966
## Dim2  0.6940001 -0.0888276
```

Assignment 1
ZWAN0168
ZHENYU WANG 550116286



Discussion on Visual Analysis 2

Design:

This study applied multidimensional scaling (MDS) to convert high-dimensional glass composition data into a two-dimensional representation using R's statistical packages, with interactive visualizations generated through the plotly library. The methodology positions glass samples based on compositional similarities, where closer spatial proximity indicates higher material resemblance. Color-coding differentiates six glass categories, while interactive features (tooltip data display, zoom controls) enable detailed inspection of individual data point coordinates and elemental attributes. A supplementary scatter plot examines relationships between selected elements through synchronized color schemes.

Evaluation:

The MDS visualization demonstrates distinct grouping patterns: glass types '1', '2', and '3' form concentrated clusters in specific regions, while types '5', '6', and '7' display more dispersed distributions, suggesting greater compositional variability. Analysis of dimension DM1 reveals strong associations between refractive index (RI) and calcium (Ca) concentrations across glass types. The RI-Ca scatter plot exhibits a positive correlation trend, with categorical differentiation evident through spatial positioning - types '1' and '2' primarily occupy central/lower regions, whereas type '7' appears in elevated zones.

Pros and Cons:

The MDS approach effectively simplifies complex compositional data into interpretable spatial relationships, facilitating rapid identification of material groupings and outliers. Interactive elements prove particularly valuable for exploring variable correlations and specimen-specific details. However, dimensionality reduction inherently obscures certain high-dimensional characteristics, potentially masking critical elemental relationships. Visualization clarity diminishes with larger datasets (>200 samples), and interactive functionalities remain inaccessible in static reporting formats.

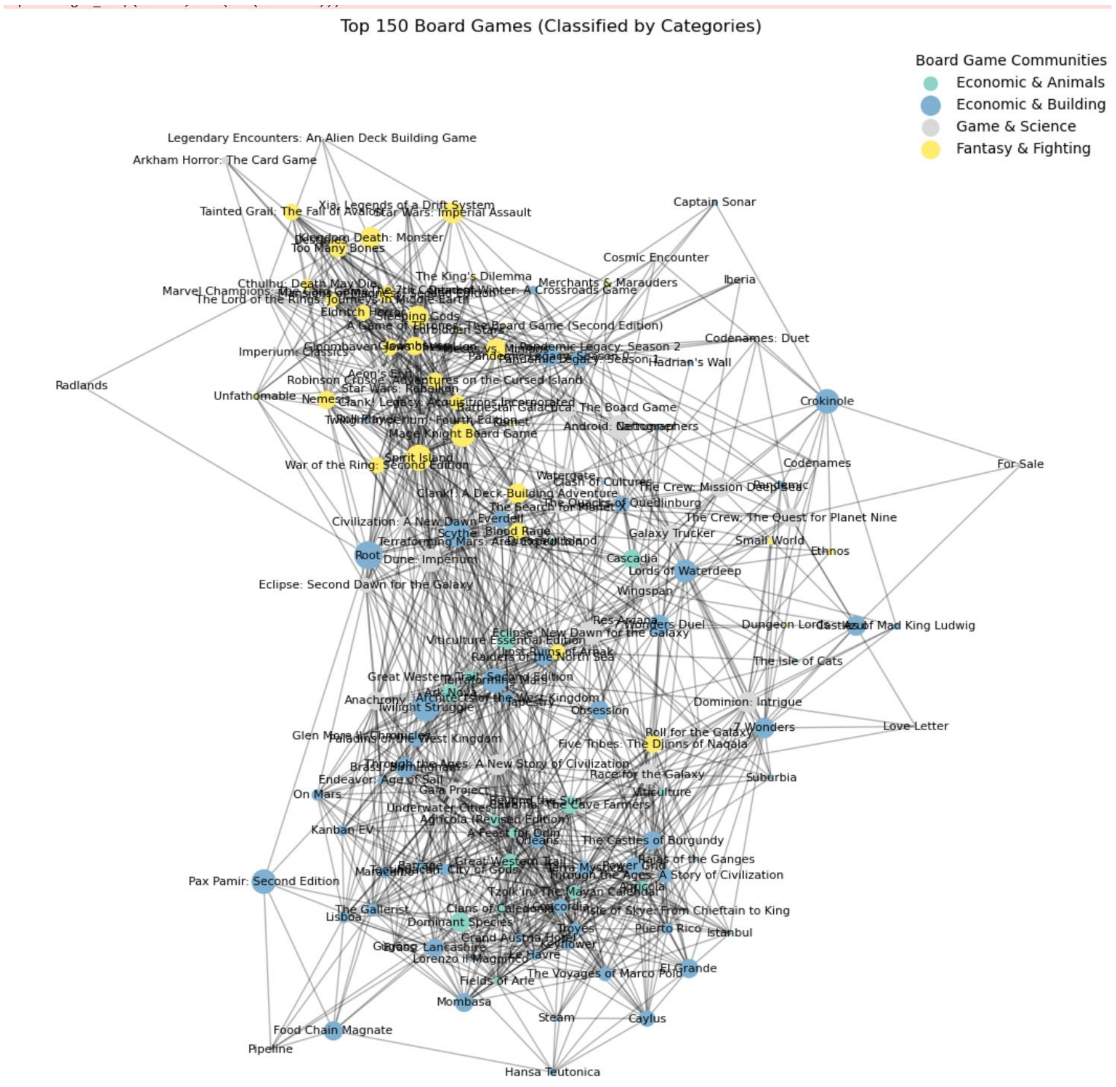
Comparison:

Parallel coordinates preserve full elemental resolution, allowing direct comparison of components like silicon and magnesium across all specimens. In contrast, MDS emphasizes structural patterns through spatial relationships but sacrifices detailed variable-level analysis. The coordinate system excels in detecting specific elemental thresholds (e.g., consistent silicon baselines), while MDS better reveals latent group similarities undetectable through axis-aligned comparisons. Their combined application enables cross-validation of findings through complementary analytical perspectives.

B1： BoardGame

Visual Analysis 1 (NetworkX):

Assignment 1
ZWAN0168
ZHENYU WANG 550116286



Discussion on Visual Analysis 1

Design:

Assignment 1
ZWAN0168
ZHENYU WANG 550116286

This analysis is carried out by using NetworkX for the board game data, first through the form of Python to filter the data, to find out the top 150 games, and then graphing, through the extraction of the game nodes in the categories of the community classification, a better display of the data of the information in the final graphs, in different colors to classify. The first two terms with the highest occurrence of categories in the relevant games in the data are used to form the community's name. This scheme performs community differentiation and Visual display of 150 top board games, link relationship and classification information can be clearly displayed.

Evaluation:

By analyzing the results of this visualization graph, which is divided into four kinds of communities, the distribution of each kind of community is obvious and there are many link nodes between the communities, which shows that there is a commonality between all these games, for example, the blue and cyan communities both have the same categories that are econometrics, but the lack of the second category is not the same between them, which is the same as the graph that shows their This is related to the fact that the graph shows that they are not the same community but are concentrated in one area, some of the games have nodes that are on the edge and don't have many connections, indicating that these games are a niche or a characteristic game.

Pros:

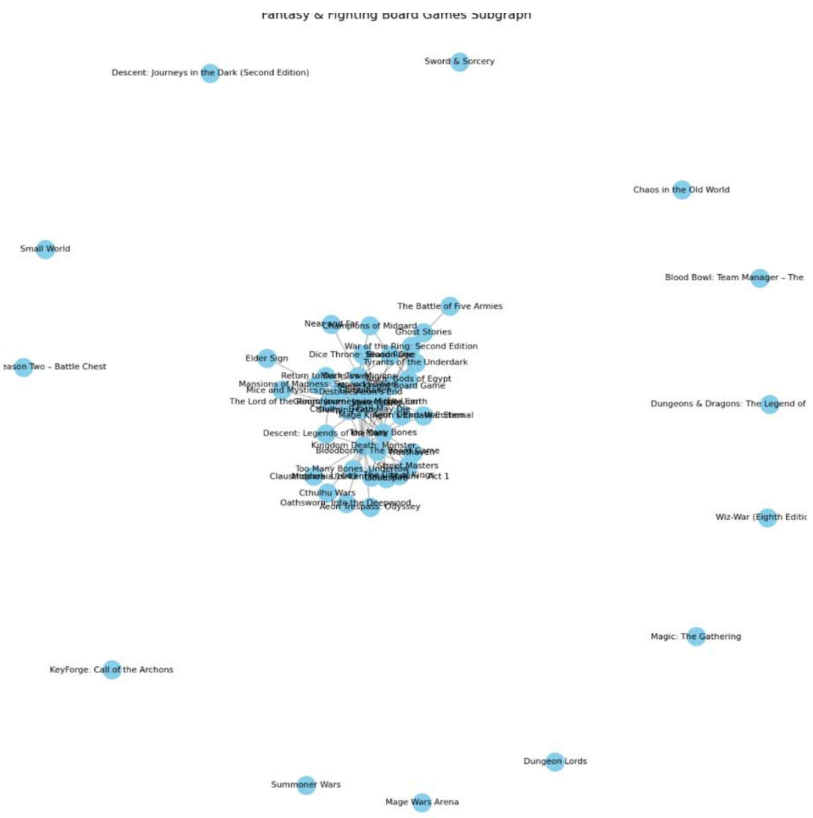
This type of analysis not only considers the results of the graph, but also the definition of the game community with categories, the information of the data can be observed effectively by classifying the data by color, and then the different size of each node can be seen as the importance of different games is different.

Cons:

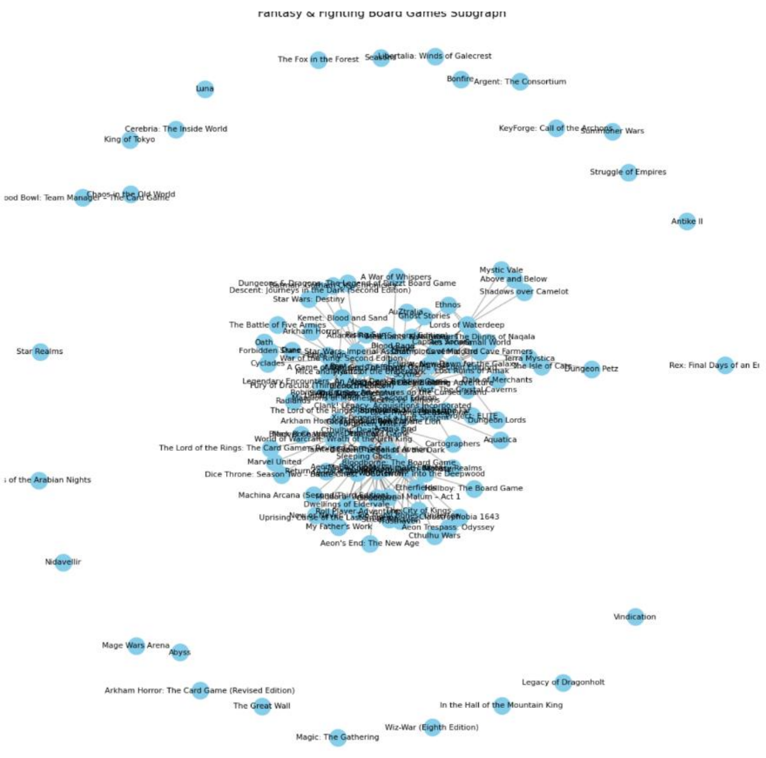
It is difficult to effectively see the details of the data in very large volumes, and large amounts of connectivity of multiple pieces of data can create visual errors in analysis, and it is not possible to effectively see the details in screenshots or reports in an interactive format.

Assignment 1
ZWAN0168
ZHENYU WANG 550116286

Visual Analysis 2:


Fantasy & Fighting Board Games Subgraph
Descent: Journeys in the Dark (Second Edition)
Sword & Sorcery
Chaos in the Old World
Small World
Blood Bowl: Team Manager – The
The Battle of Five Armies
Elder Sign
Dungeons & Dragons: The Legend of
Wiz-War (Eighth Editi
Magic: The Gathering
KeyForge: Call of the Archons
Dungeon Lords
Summoner Wars
Mage Wars Arena

1The board games have both Fantasy and Fighting



2The board games have Fantasy or Fighting

Assignment 1
ZWAN0168
ZHENYU WANG 550116286

Discussion on Visual Analysis 2

Design:

Four communities were derived from the first excerpt of 150 games, and one of them, fantasy and fighting, was selected for the second analysis. This data processing was carried out through NetworkX to read the GML file's data, filter the nodes to construct subgraphs, and visualize the data through matplotlib. The screening method is divided into two for comparing the connection between the games, one is the game with both fantasy and fighting, and the other is the game with either fantasy or fighting, and due to the uniformity of the selected communities, it is not classified by color.

Evaluation:

In the first graph (3The board games have both Fantasy and Fighting), with a smaller selection of categories with a relatively small amount of data, the correlation between the games can be clearly seen, as well as the anomalous part, where a small number of games are scattered around the collective and are not connected to the other nodes, suggesting that these games are weakly correlated with the other games.

In the second graph (4The board games have Fantasy or Fighting) Shown in this graph are games that fit into one of the categories, so this data will have more games present, and it can also be seen that there is a strong correlation between these games converging in the center, as well as a handful of games that have specificity in not having any nodes connected around them.

Pro:

The filtering conditions are clear and allow for effective analysis of the target data, and the structure of the graphs allows for analysis of the core group of gamers

Con:

The compact structure in the middle, scattered around without connecting lines, and the small number of nodes, can lead to some nodes not being connected due to filtering losing a node in the middle that possesses a connection, and failing to recognize the connection of the game across categories.

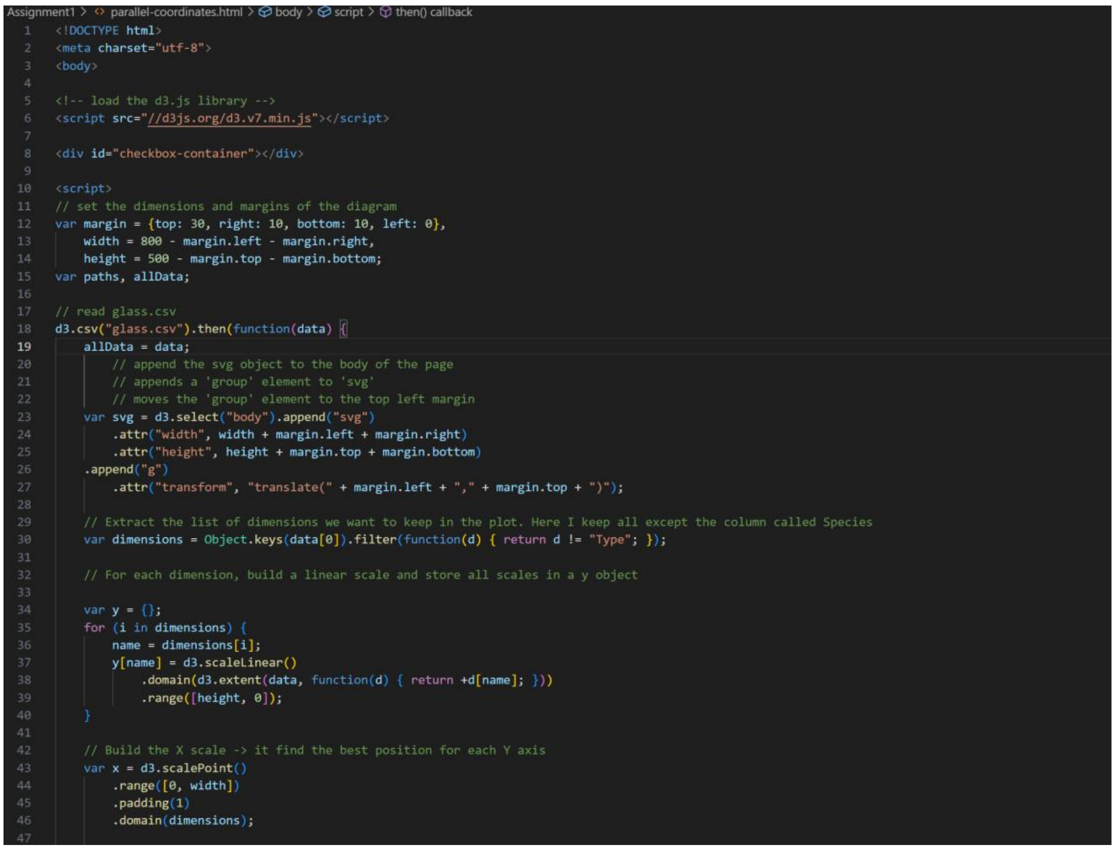
Comparison:

Comparing the two kinds of Visual Analysis respectively by even a different direction of analysis, the first kind of graph through the screening of the first 150 games to classify, that can see part of the game through the number of connections and node densities to see which games in the graph of the most influential, through the use of color classification and the size of the node can be the structure of the hierarchy is clear, and the second kind of by selecting the categories to filter the data to do analysis can see the correlation between the games of a particular topic, but also found that in some communities there are anomalies, these nodes are often not connected to any other nodes. The second will lead to unstable data volume according to the filtered categories.

Assignment 1
ZWAN0168
ZHENYU WANG 550116286

Appendix:

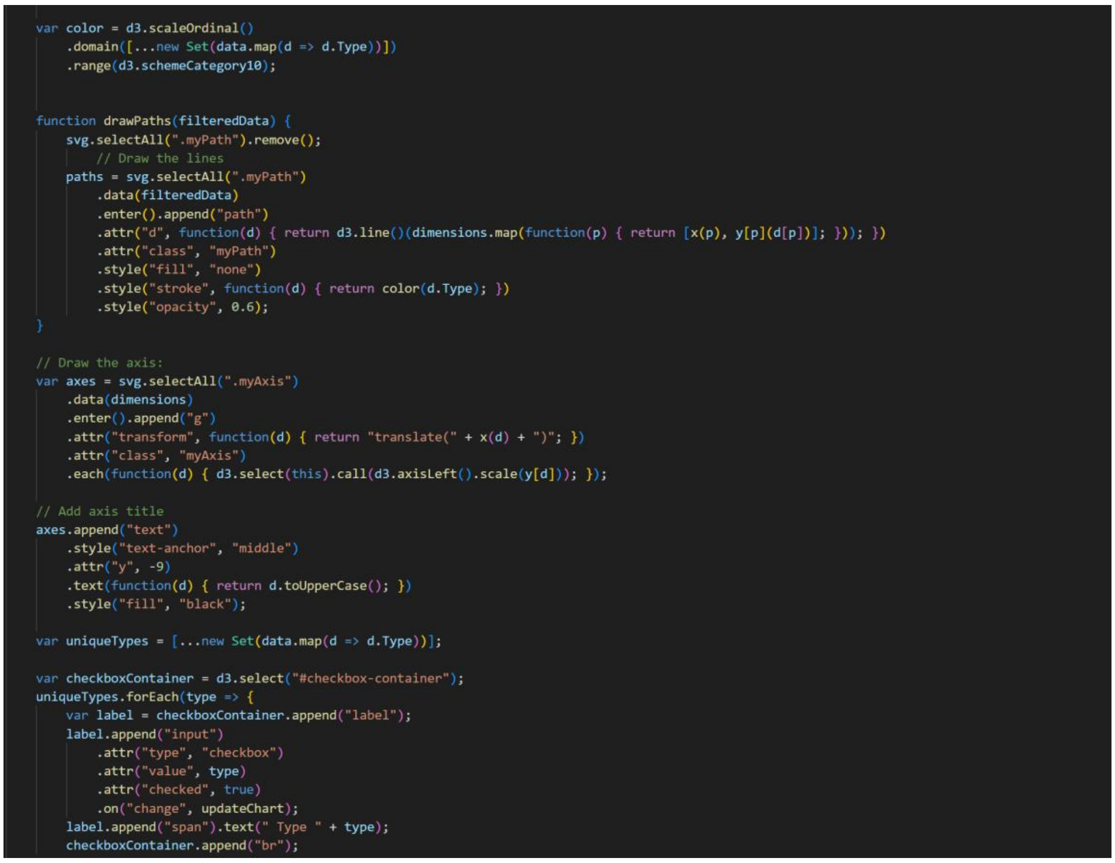
Code:



Assignment1 > parallel-coordinates.html > body > script > then() callback

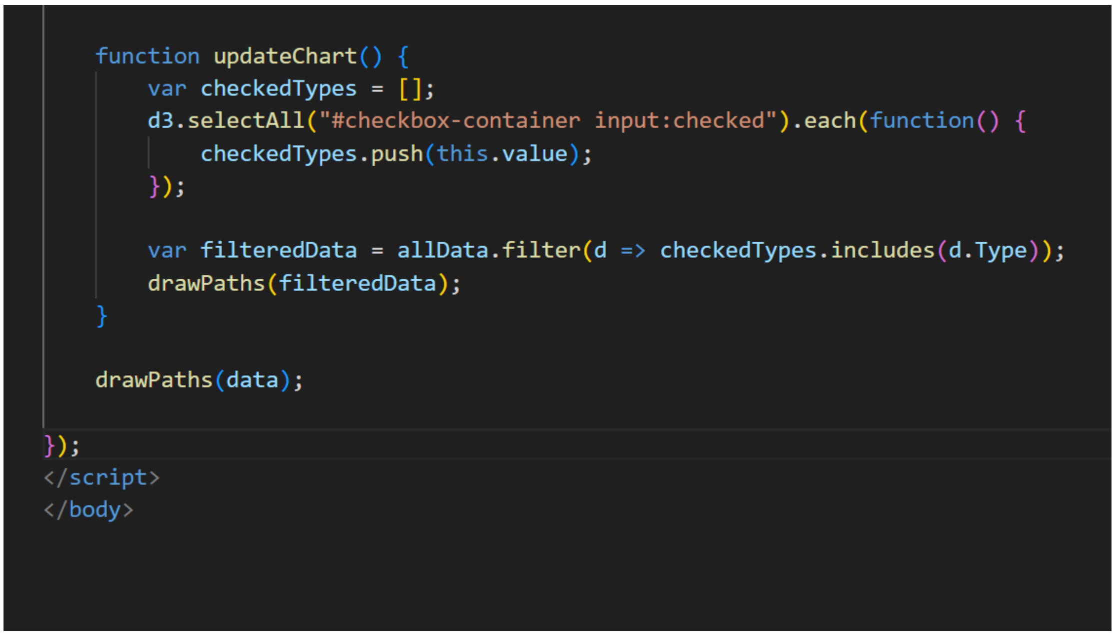
```
<!DOCTYPE html>
<meta charset="utf-8">
<body>

<!-- load the d3.js library -->
<script src="//d3js.org/d3.v7.min.js"></script>

<div id="checkbox-container"></div>

<script>
// set the dimensions and margins of the diagram
var margin = {top: 30, right: 10, bottom: 10, left: 0},
    width = 800 - margin.left - margin.right,
    height = 500 - margin.top - margin.bottom;
var paths, allData;

// read glass.csv
d3.csv("glass.csv").then(function(data) {
    allData = data;
        // append the svg object to the body of the page
        // appends a 'group' element to 'svg'
        // moves the 'group' element to the top left margin
    var svg = d3.select("body").append("svg")
        .attr("width", width + margin.left + margin.right)
        .attr("height", height + margin.top + margin.bottom)
    .append("g")
        .attr("transform", "translate(" + margin.left + "," + margin.top + ")");

    // Extract the list of dimensions we want to keep in the plot. Here I keep all except the column called Species
    var dimensions = Object.keys(data[0]).filter(function(d) { return d != "Type"; });

    // For each dimension, build a linear scale and store all scales in a y object

    var y = {};
    for (i in dimensions) {
        name = dimensions[i];
        y[name] = d3.scaleLinear()
            .domain(d3.extent(data, function(d) { return +d[name]; }))
            .range([height, 0]);
    }

    // Build the X scale -> it find the best position for each Y axis
    var x = d3.scalePoint()
        .range([0, width])
        .padding(1)
        .domain(dimensions);

```

Assignment 1
ZWAN0168
ZHENYU WANG 550116286



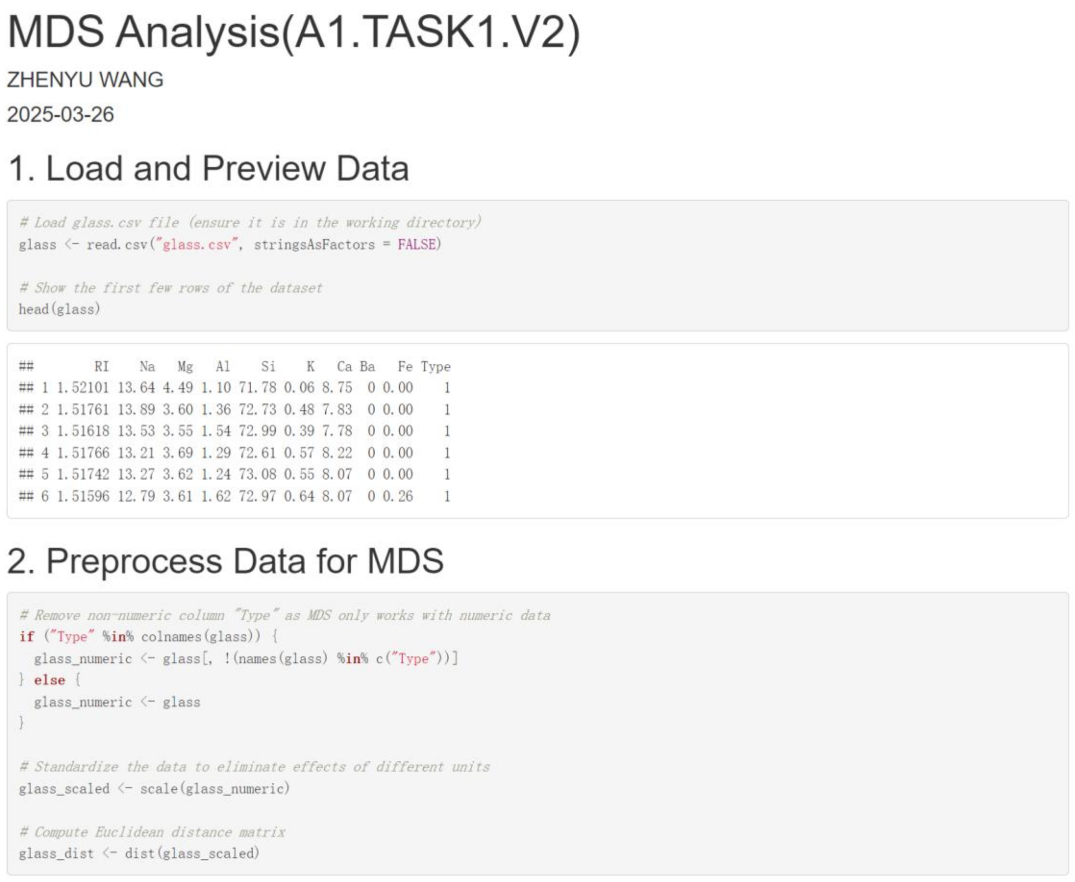
```
var color = d3.scaleOrdinal()
    .domain([...new Set(data.map(d => d.Type))])
    .range(d3.schemeCategory10);


function drawPaths(filteredData) {
    svg.selectAll(".myPath").remove();
        // Draw the lines
    paths = svg.selectAll(".myPath")
        .data(filteredData)
        .enter().append("path")
        .attr("d", function(d) { return d3.line()(dimensions.map(function(p) { return [x(p), y[p](d[p])]; })); })
        .attr("class", "myPath")
        .style("fill", "none")
        .style("stroke", function(d) { return color(d.Type); })
        .style("opacity", 0.6);
}

// Draw the axis:
var axes = svg.selectAll(".myAxis")
    .data(dimensions)
    .enter().append("g")
    .attr("transform", function(d) { return "translate(" + x(d) + ")"; })
    .attr("class", "myAxis")
    .each(function(d) { d3.select(this).call(d3.axisLeft().scale(y[d])); });

// Add axis title
axes.append("text")
    .style("text-anchor", "middle")
    .attr("y", -9)
    .text(function(d) { return d.toUpperCase(); })
    .style("fill", "black");

var uniqueTypes = [...new Set(data.map(d => d.Type))];

var checkboxContainer = d3.select("#checkbox-container");
uniqueTypes.forEach(type => {
    var label = checkboxContainer.append("label");
    label.append("input")
        .attr("type", "checkbox")
        .attr("value", type)
        .attr("checked", true)
        .on("change", updateChart);
    label.append("span").text(" Type " + type);
    checkboxContainer.append("br");
```

5CODE FOR VA1



```
    function updateChart() {
        var checkedTypes = [];
        d3.selectAll("#checkbox-container input:checked").each(function() {
            checkedTypes.push(this.value);
        });

        var filteredData = allData.filter(d => checkedTypes.includes(d.Type));
        drawPaths(filteredData);
    }

    drawPaths(data);

});
</script>
</body>
```

Assignment 1
ZWAN0168
ZHENYU WANG 550116286



MDS Analysis(A1.TASK1.V2)

ZHENYU WANG

2025-03-26

1. Load and Preview Data

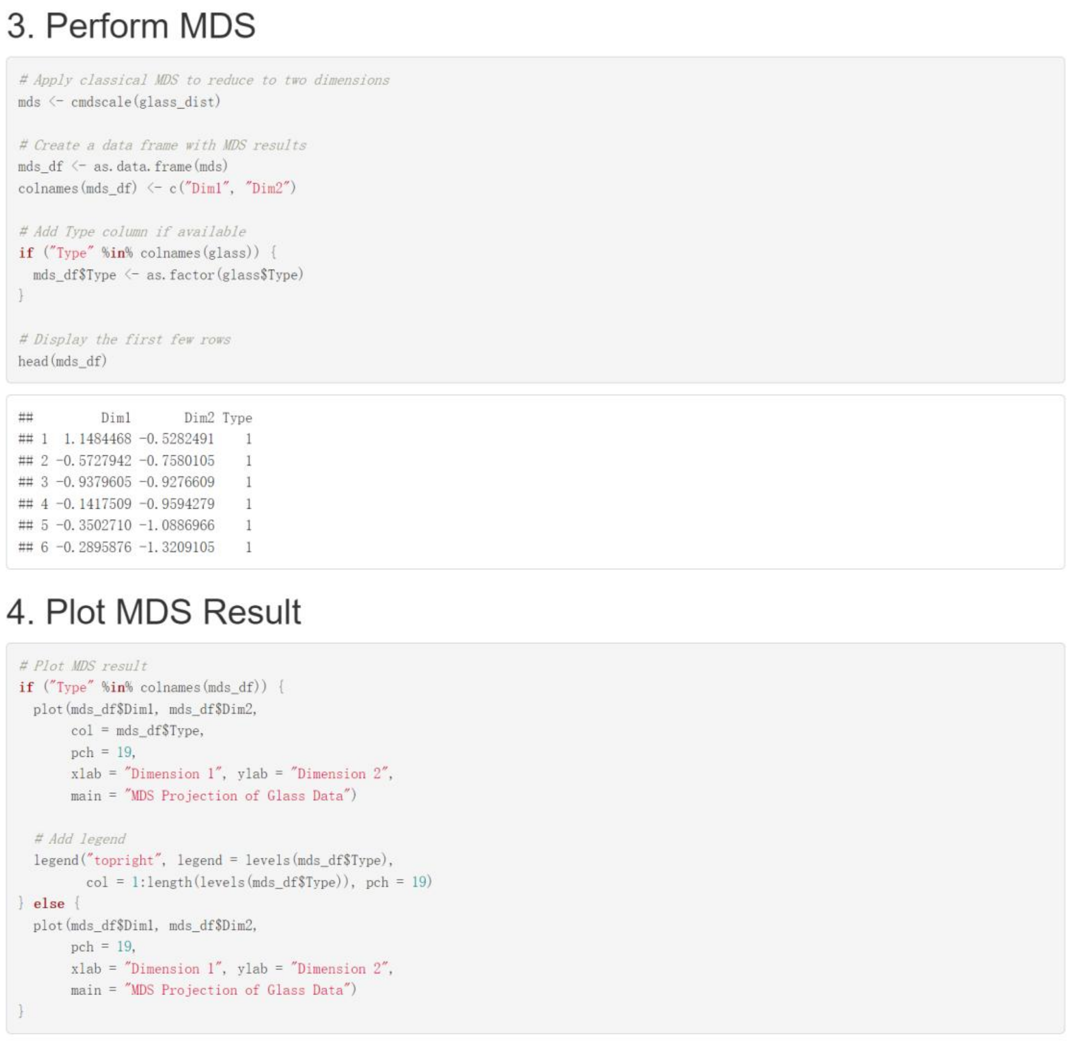
```
# Load glass.csv file (ensure it is in the working directory)
glass <- read.csv("glass.csv", stringsAsFactors = FALSE)

# Show the first few rows of the dataset
head(glass)
```

```
##        RI    Na   Mg   Al    Si    K   Ca Ba   Fe Type
## 1 1.52101 13.64 4.49 1.10 71.78 0.06 8.75  0 0.00    1
## 2 1.51761 13.89 3.60 1.36 72.73 0.48 7.83  0 0.00    1
## 3 1.51618 13.53 3.55 1.54 72.99 0.39 7.78  0 0.00    1
## 4 1.51766 13.21 3.69 1.29 72.61 0.57 8.22  0 0.00    1
## 5 1.51742 13.27 3.62 1.24 73.08 0.55 8.07  0 0.00    1
## 6 1.51596 12.79 3.61 1.62 72.97 0.64 8.07  0 0.26    1
```

2. Preprocess Data for MDS

```
# Remove non-numeric column "Type" as MDS only works with numeric data
if ("Type" %in% colnames(glass)) {
  glass_numeric <- glass[, !(names(glass) %in% c("Type"))]
} else {
  glass_numeric <- glass
}

# Standardize the data to eliminate effects of different units
glass_scaled <- scale(glass_numeric)

# Compute Euclidean distance matrix
glass_dist <- dist(glass_scaled)
```

Assignment 1
ZWAN0168
ZHENYU WANG 550116286



3. Perform MDS

```
# Apply classical MDS to reduce to two dimensions
mds <- cmdscale(glass_dist)

# Create a data frame with MDS results
mds_df <- as.data.frame(mds)
colnames(mds_df) <- c("Dim1", "Dim2")

# Add Type column if available
if ("Type" %in% colnames(glass)) {
  mds_df$Type <- as.factor(glass$Type)
}

# Display the first few rows
head(mds_df)
```

```
##         Dim1       Dim2 Type
## 1  1.1484468 -0.5282491    1
## 2 -0.5727942 -0.7580105    1
## 3 -0.9379605 -0.9276609    1
## 4 -0.1417509 -0.9594279    1
## 5 -0.3502710 -1.0886966    1
## 6 -0.2895876 -1.3209105    1
```

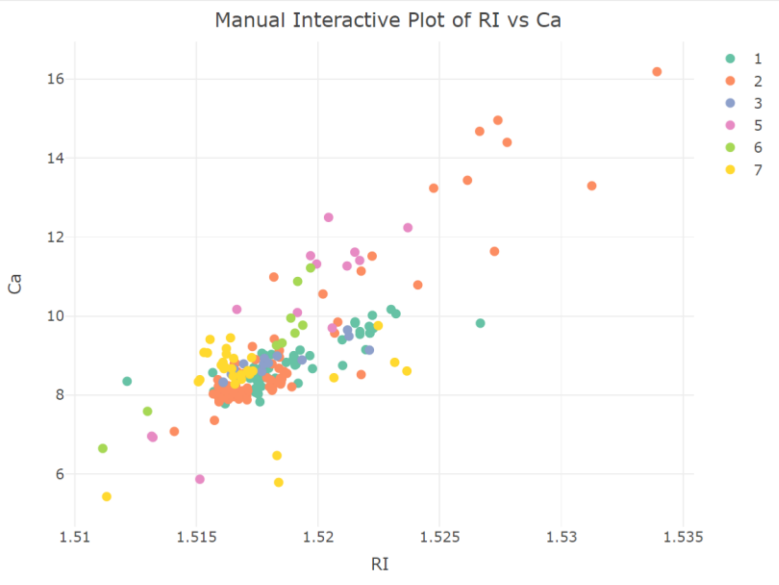
4. Plot MDS Result

```
# Plot MDS result
if ("Type" %in% colnames(mds_df)) {
  plot(mds_df$Dim1, mds_df$Dim2,
       col = mds_df$Type,
       pch = 19,
       xlab = "Dimension 1", ylab = "Dimension 2",
       main = "MDS Projection of Glass Data")

  # Add legend
  legend("topright", legend = levels(mds_df$Type),
         col = 1:length(levels(mds_df$Type)), pch = 19)
} else {
  plot(mds_df$Dim1, mds_df$Dim2,
       pch = 19,
       xlab = "Dimension 1", ylab = "Dimension 2",
       main = "MDS Projection of Glass Data")
}
```

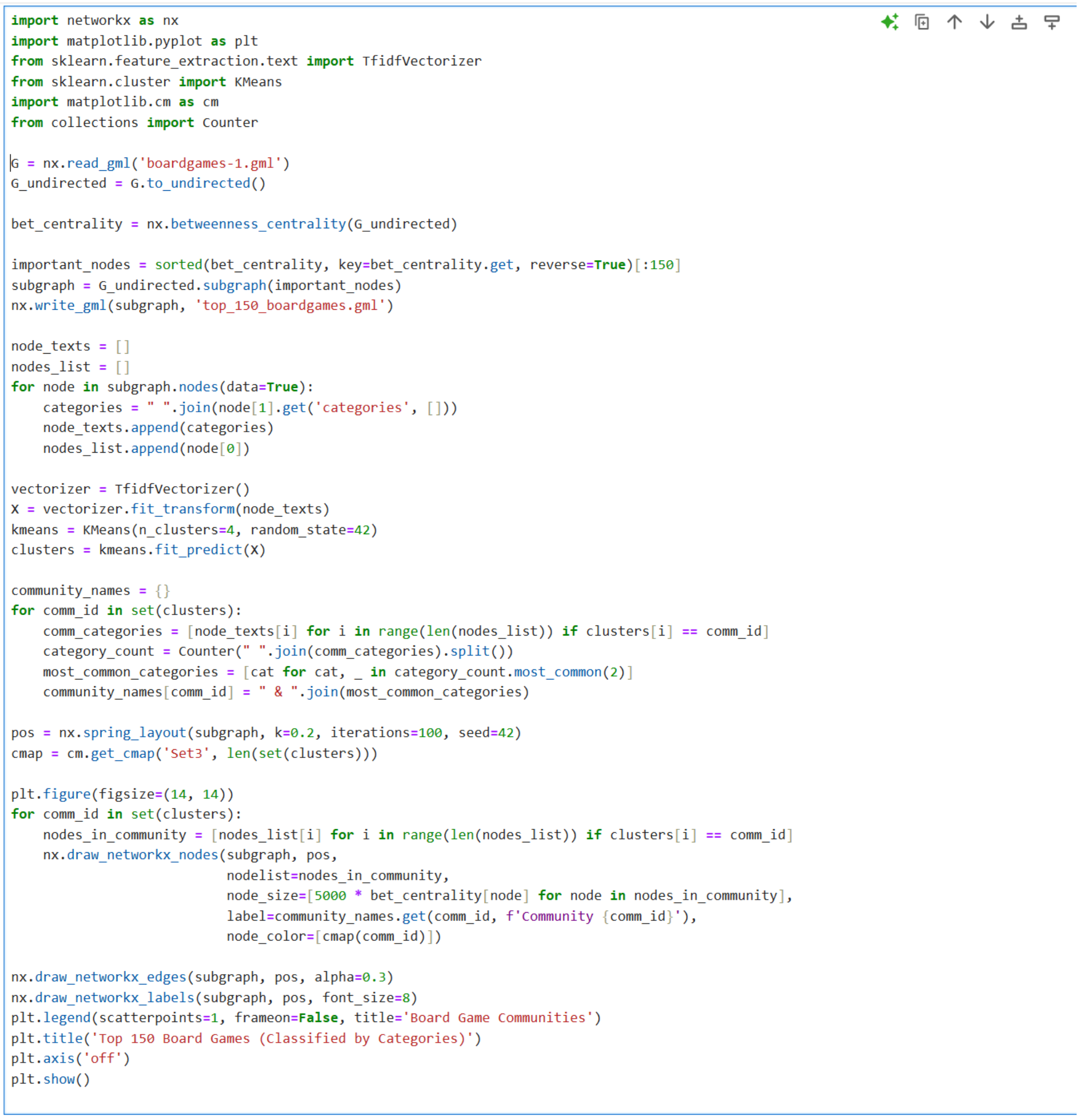
Assignment 1
ZWAN0168
ZHENYU WANG 550116286

5. Two Variables for Comparison

```r
manual_var_x <- "RI"
manual_var_y <- "Ca"

# Build data and plot
manual_df <- glass[, c(manual_var_x, manual_var_y)]
if ("Type" %in% colnames(glass)) {
  manual_df$Type <- as.factor(glass$Type)
}

# Plot interactive scatterplot using plotly
plot_ly(manual_df, x = ~get(manual_var_x), y = ~get(manual_var_y), color = ~Type,
        type = "scatter", mode = "markers",
        marker = list(size = 8)) %>%
  layout(title = paste("Manual Interactive Plot of", manual_var_x, "vs", manual_var_y),
         xaxis = list(title = manual_var_x),
         yaxis = list(title = manual_var_y))
```



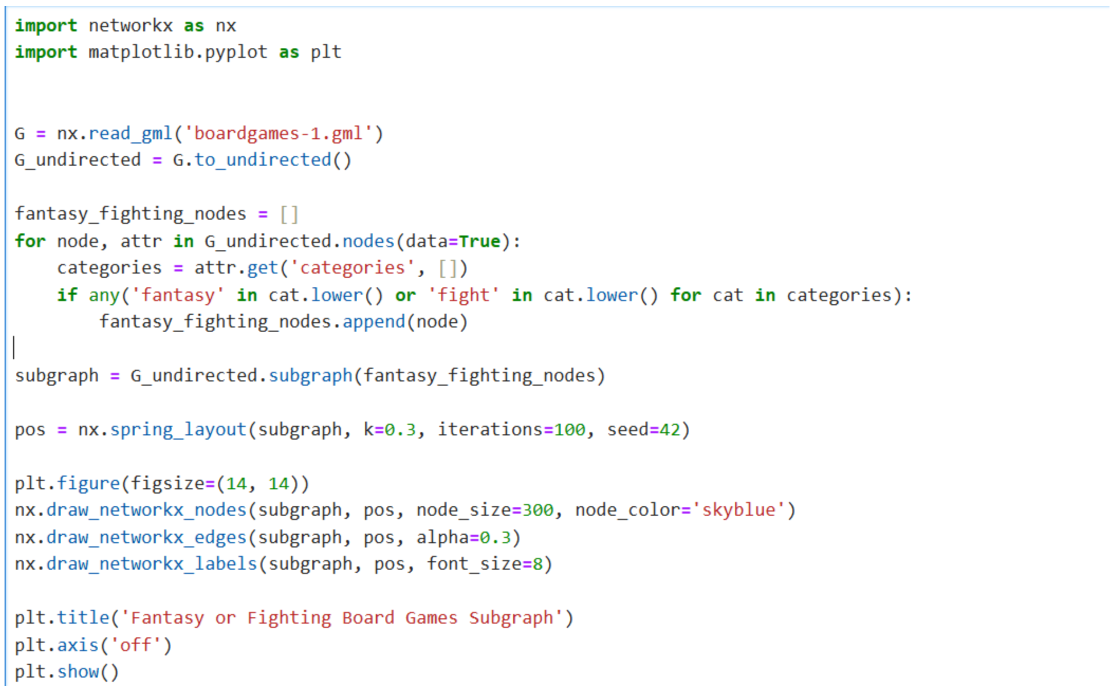
Assignment 1
ZWAN0168
ZHENYU WANG 550116286



```python
import networkx as nx
import matplotlib.pyplot as plt
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans
import matplotlib.cm as cm
from collections import Counter

G = nx.read_gml('boardgames-1.gml')
G_undirected = G.to_undirected()

bet_centrality = nx.betweenness_centrality(G_undirected)

important_nodes = sorted(bet_centrality, key=bet_centrality.get, reverse=True)[:150]
subgraph = G_undirected.subgraph(important_nodes)
nx.write_gml(subgraph, 'top_150_boardgames.gml')

node_texts = []
nodes_list = []
for node in subgraph.nodes(data=True):
    categories = " ".join(node[1].get('categories', []))
    node_texts.append(categories)
    nodes_list.append(node[0])

vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(node_texts)
kmeans = KMeans(n_clusters=4, random_state=42)
clusters = kmeans.fit_predict(X)

community_names = {}
for comm_id in set(clusters):
    comm_categories = [node_texts[i] for i in range(len(nodes_list)) if clusters[i] == comm_id]
    category_count = Counter(" ".join(comm_categories).split())
    most_common_categories = [cat for cat, _ in category_count.most_common(2)]
    community_names[comm_id] = " & ".join(most_common_categories)

pos = nx.spring_layout(subgraph, k=0.2, iterations=100, seed=42)
cmap = cm.get_cmap('Set3', len(set(clusters)))

plt.figure(figsize=(14, 14))
for comm_id in set(clusters):
    nodes_in_community = [nodes_list[i] for i in range(len(nodes_list)) if clusters[i] == comm_id]
    nx.draw_networkx_nodes(subgraph, pos,
                           nodelist=nodes_in_community,
                           node_size=[5000 * bet_centrality[node] for node in nodes_in_community],
                           label=community_names.get(comm_id, f'Community {comm_id}'),
                           node_color=[cmap(comm_id)])

nx.draw_networkx_edges(subgraph, pos, alpha=0.3)
nx.draw_networkx_labels(subgraph, pos, font_size=8)
plt.legend(scatterpoints=1, frameon=False, title='Board Game Communities')
plt.title('Top 150 Board Games (Classified by Categories)')
plt.axis('off')
plt.show()
```

6Code for B1 V1

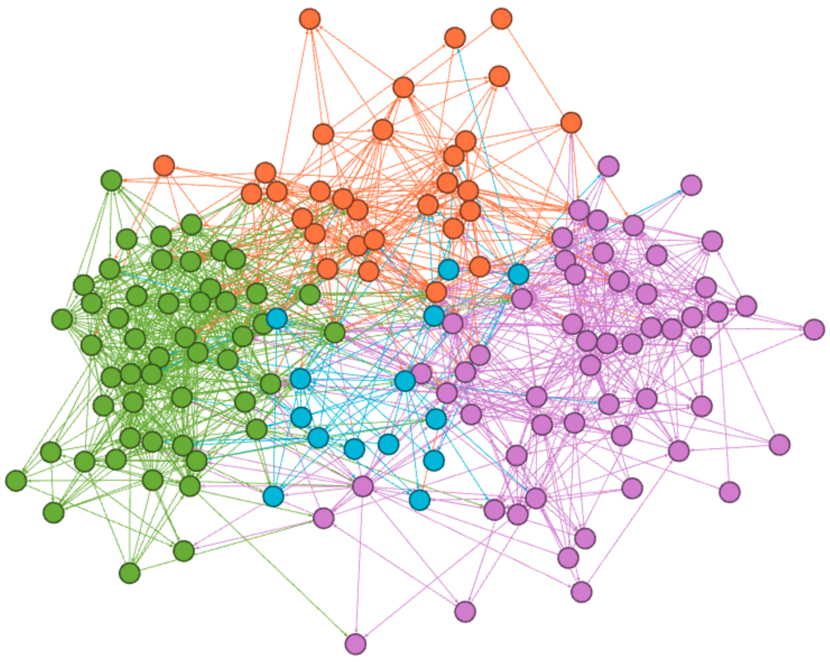
Assignment 1
ZWAN0168
ZHENYU WANG 550116286



```python
import networkx as nx
import matplotlib.pyplot as plt


G = nx.read_gml('boardgames-1.gml')
G_undirected = G.to_undirected()

fantasy_fighting_nodes = []
for node, attr in G_undirected.nodes(data=True):
    categories = attr.get('categories', [])
    if any('fantasy' in cat.lower() or 'fight' in cat.lower() for cat in categories):
        fantasy_fighting_nodes.append(node)

subgraph = G_undirected.subgraph(fantasy_fighting_nodes)

pos = nx.spring_layout(subgraph, k=0.3, iterations=100, seed=42)

plt.figure(figsize=(14, 14))
nx.draw_networkx_nodes(subgraph, pos, node_size=300, node_color='skyblue')
nx.draw_networkx_edges(subgraph, pos, alpha=0.3)
nx.draw_networkx_labels(subgraph, pos, font_size=8)

plt.title('Fantasy or Fighting Board Games Subgraph')
plt.axis('off')
plt.show()
```

CODE FOR B1 V2

The other visualization method for B1 BOARD GAME TOP150 GAMES:



Assignment 1
ZWAN0168
ZHENYU WANG 550116286

Design:

The visualization employs Gephi's force-directed layout using the Yifan Hu algorithm, which simulates attraction/repulsion forces between nodes to achieve natural spatial organization. Community detection is implemented through the Modularity algorithm (Louvain method), automatically grouping nodes based on connection density. Chromatic encoding distinguishes communities, while the design integrates three core functionalities: force simulation for node arrangement, data-driven community identification, and visual cluster differentiation.

Evaluation:

This visualization identifies four primary communities with observable inter-cluster connections, suggesting shared design characteristics. Densely connected regions indicate close relationships within the group, while bridging nodes imply mixed game mechanics. Peripheral nodes with limited connections correspond to niche or experimental designs. Spatial node locations reflect design similarities, despite the lack of absolute coordinate references.

Pros

It can lay out a clear presentation of complex associations, identify groups based on the density of connections, distinguish the characteristics of different data by color and size, and flexibly adjust node attributes for unused analyses.

Cons

The visualization method suffers from the following core limitations: overlapping nodes in high-density regions make it difficult to identify details; static output loses interactive features (e.g., hover information); community delineation relies on algorithmic parameters and requires additional validation of practical significance; and the lack of a quantitative coordinate system limits the precise analysis of spatial relationships.

Comparison

This graphical analysis is methodologically complementary to the NetworkX implementation: NetworkX uses predefined taxonomic labels (e.g., “economic strategies”) and fixed color coding, focusing on presentation rather than structural discovery, which improves labeling control of the report outputs but does not allow for automated identification of potential communities, and Gephi, which uses connectivity density analysis to enable detection of organic communities, which is superior to exploratory structural analysis, whereas NetworkX has the advantage of reproducible, standardized outputs. NetworkX is superior in reproducible standardized output. Both are suitable for different stages of the analytical process - Gephi for initial structural exploration, and NetworkX for final presentation of results.

Assignment 1
ZWAN0168
ZHENYU WANG 550116286

Reference:

Shakhgildyan, G., Lipatiev, A., Lotarev, S., Fedotov, S., & Sigaev, V. (2020). Glass: Home of the Periodic Table. *Frontiers in chemistry*, *8*, 384. https://doi.org/10.3389/fchem.2020.00384

AI Tools Used

ChatGPT: Used for graph fix, modify the code based on the course learning and add interactive functions.

DeepL: Translate some special words.