Contents

DSA5101 Project: Young People Survey – Findings on Hobbies & Interests

Introduction

The purpose of this study is to explore the hobbies and interests of young people. The survey was done in 2013 with over 1000 participants, from Comenius University Bratislava. They are all Slovakian nationality, aged between 15-30. By running the survey results through some algorithms, we seek to attain some findings with regards to the hobbies and interests of young people.

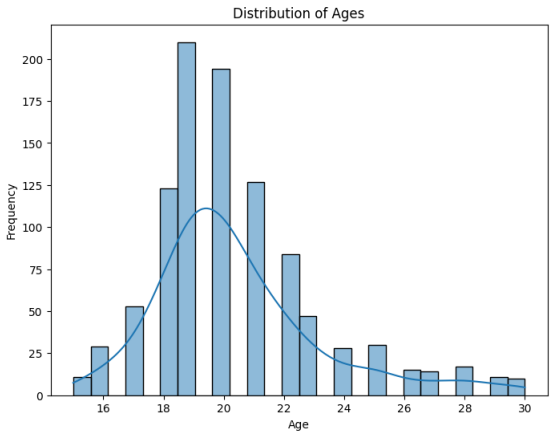
In recent times, we are increasingly connected online but we are ironically increasingly disconnected in real life. With this trend in the backdrop, we are thinking that data techniques/algorithms and their findings could be useful to help young people find other like-minded people and recommend them new hobbies that they can engage in to find fulfilment.

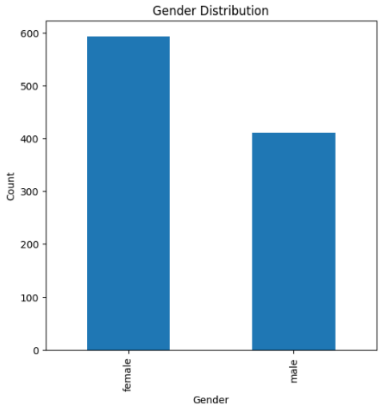
In this notebook, we will do so with the following algorithms:

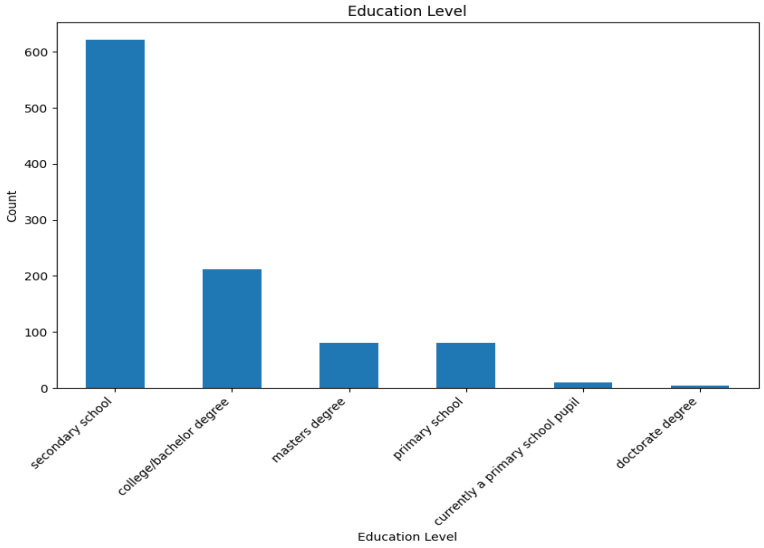
- K Means Clustering with dimensionality reduction - the outcome of this machine learning is to group young people with overlaps of multiple hobby groups
- Graphing and graph clustering algorithms - the outcome of this machine learning is to give suggestions to young people of what other hobbies, which could be within the same hobby cluster, that can be considered to find new friends.

Data Exploration

We first explored the data within the original data set, starting with their basic information like gender, age and education level.







As observed from the graphs above, we can tell that most of the survey participants are concentrated in the age range of between 18 to 22 years old in the wider range of 15 to 30 years old. The gender ratio is also around 60 – 40% between female and male. Finally, given that these survey participants are mainly current college students at the time of the survey, the highest level of education is secondary school.

Truncating and preprocessing of dataset

We looked at the csv file of the survey responses and identified that we wanted to look into hobbies and interests as we mentioned in the introduction. Therefore, we proceed to only take the columns which are associated with hobbies and interests. In the csv file, we took

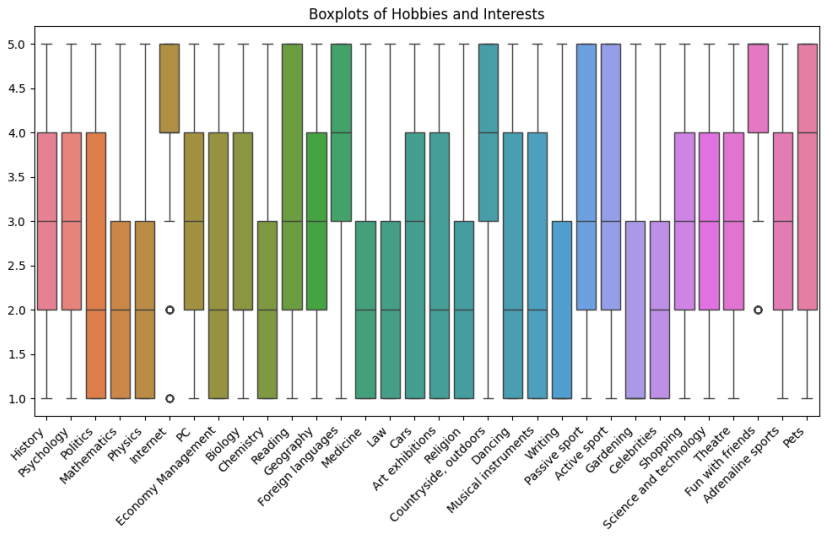
32 columns under the hobbies and interests' section. We then proceeded to fill the empty cells with the median of the entire column. This step is shown in the code snippet below, along with a code segment to copy the hobbies dataframe:

```
# Select columns which are Hobbies and Interests: 'History' to 'Pets'
(inclusive)
hobbies_df = df.loc[:, 'History':'Pets']

# fill the na rows in each column in hobbies dataframe with the median of
each column
for col in hobbies_df.columns:
    hobbies_df[col] = hobbies_df[col].fillna(hobbies_df[col].median())

hobbies = hobbies_df.copy()
```

Further Data Exploration – Hobbies & Interests Dataset

In this section, we further explore the data within the truncated dataset. We use simpler statistical methods to understand the data and think ahead about the possible algorithms we can employ to reach novel findings and deeper insights which might be useful for young people.

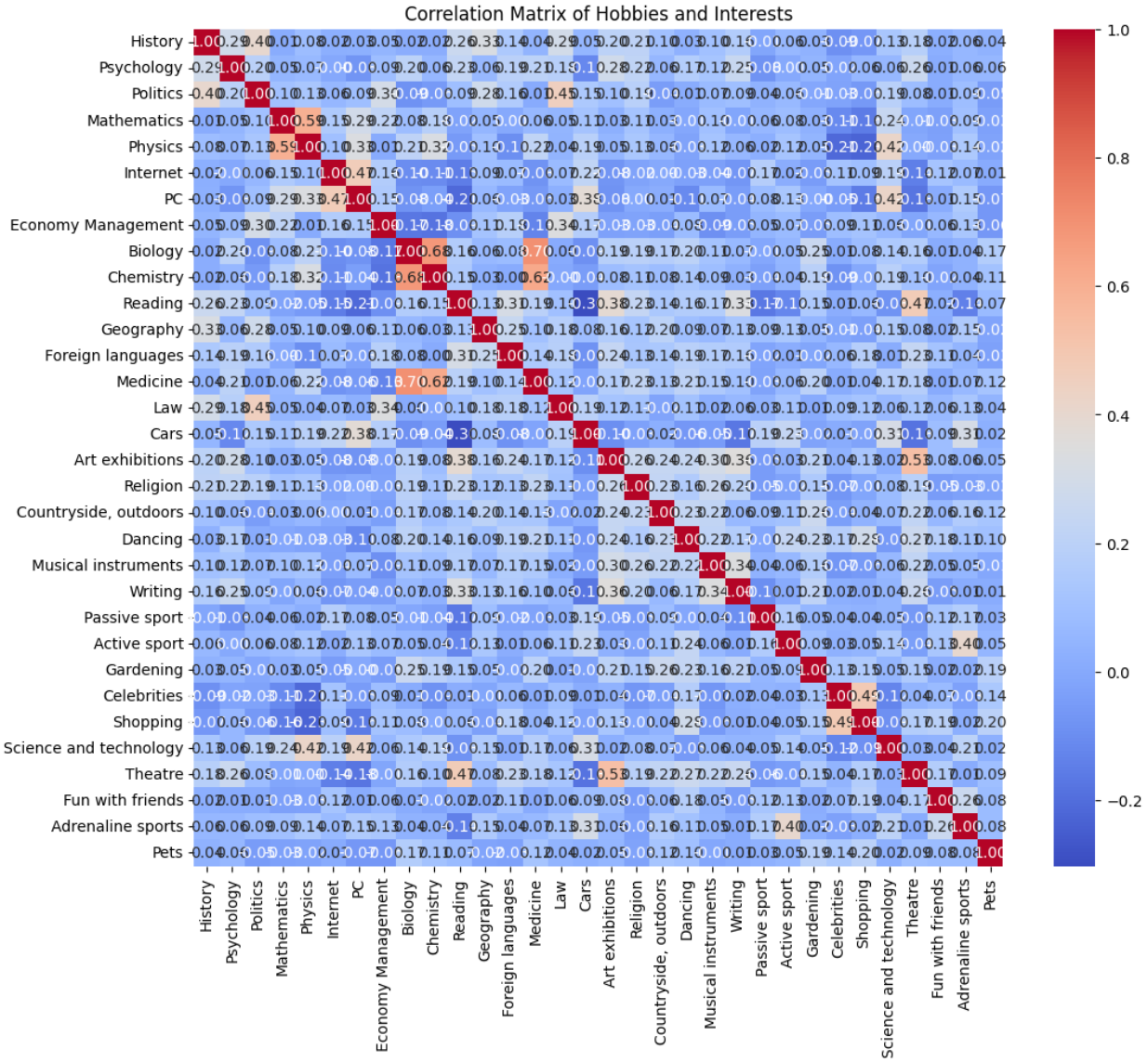
Since the survey responses were requested on a Likert scale, we first used a boxplot to visualize and get a rough idea of the spread of values which the participants responded with. The figure below shows the results:



Through the boxplots, we notice that most responses lie within 2 to 4. This is a typical range where we expect to find the responses for the Likert Scale type survey questions. We noticed

that ‘Internet’ and ‘Fun with friends’ is very skewed towards high scores. We took a mental note here that leaving these 2 features in might not be the best option.

Next, we also wanted to check if we could find any meaningful correlations between hobbies i.e. this would mean that if someone expresses interest in one hobby, a higher correlation with another hobby, above a certain threshold, would mean that same person might also like the other hobby. We presented the correlations using a heatmap of the correlation matrix of the hobbies as shown below:



	Hobby 1	Hobby 2	Correlation
0	Medicine	Biology	0.702746
1	Biology	Chemistry	0.677562
2	Medicine	Chemistry	0.621256
3	Physics	Mathematics	0.593086
4	Art exhibitions	Theatre	0.533017
5	Shopping	Celebrities	0.485098
6	Reading	Theatre	0.473168
7	Internet	PC	0.467300
8	Law	Politics	0.450711
9	PC	Science and technology	0.420512

From the correlation matrix, we can tell that most of the hobbies are not very highly correlated. Hence, it is not immediately clear which are the groupings to be made. However, from the table of the top 10 pairings with highest correlation scores, we can at least tell that hobbies which are similar in nature will tend to have a higher correlation. The top 4 pairings can all be categorized under Mathematics and Sciences. We will come back to revisit this idea later in the report.

After the data explorations done above, we cannot conclusively remove any of the hobbies (features), except for ‘Internet’ and ‘Fun with friends’ for the following reasons: (i) from the boxplot, we can tell that most of the participants responded to them with a high score which results in low variance and (ii) they are general activities which people engage in and do not belong to any domain of hobbies and interests. We will hence proceed to drop the 2 columns from the data frame as follows:

```
# Remove 'Internet' and 'Fun with friends' from the hobbies DataFrame
hobbies = hobbies.drop(columns=['Internet', 'Fun with friends'])
```

Clustering using K-means algorithm

To briefly recap on the algorithm, the main idea is to identify some representatives of each cluster ("cluster centers") and assign each data point to its closest datapoint center in terms of Euclidean distance. We perform these 2 steps iteratively until a stopping criterion is reached.

More concretely, these are the steps which were taken to run the K-mean algorithm (along with their code snippets):

1. **Scaling the data:** this was done to ensure that the different features contribute equally to the model. This step is not particularly important for this Likert Scale type data but since the range of responses might slightly vary, there is no drawback in performing standard scaling to all the hobbies' ratings (features)

```
# Scale the data
scaler = StandardScaler()
scaled_hobbies = scaler.fit_transform(hobbies)
```

2. **Initialization for K-means:** K-means ++ was employed as the initialization method. This was chosen to: (i) improve initialization by ensuring that the initial centroids selected are well spread out across the dataset, which leads to better clustering results compared to random initialization and (ii) avoid converging to suboptimal solutions (local minima), which can occur if initial centroids are too close to each other or poorly positioned
3. **Elbow method:** this was the technique we used to determine the optimal number of clusters where adding more clusters provides diminishing returns

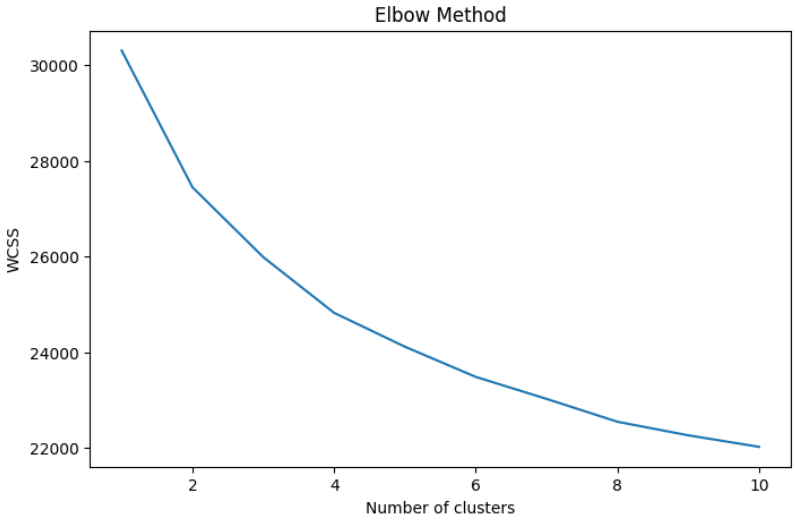
```
# Elbow method and K-means++ initialization
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)
    kmeans.fit(scaled_hobbies)
    wcss.append(kmeans.inertia_)
```

4. **Silhouette analysis:** is a method used to evaluate the quality of the clusters generated by the algorithm. A Silhouette score of the outcome of the clustering algorithm will give us a numerical indication of how well-separate the clusters are

```
# Silhouette analysis
silhouette_scores = []
for i in range(2, 11):  # Silhouette score is not defined for 1 cluster
```

```
kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)
kmeans_labels = kmeans.fit_predict(scaled_hobbies)
silhouette_avg = silhouette_score(scaled_hobbies, kmeans_labels)
silhouette_scores.append(silhouette_avg)
```

The clustering algorithm (as outlined in the above steps) produced 4 clusters. The elbow method is used to identify 4 as the optimal number cluster, where the curve starts to flatten out as below:



Showing them in a key and value format: {Cluster #: # of respondents} = {0: 174, 1: 265, 2: 316, 3:265}. The silhouette score for this clustering is only **0.074**. To give a gauge on how a good silhouette score is perceived, we follow this intuition below:

- Above 0.5: indicates that clusters are well-defined and separated;
- 0.3 to 0.5: Indicates that the clusters may be overlapping but can still be considered reasonably well-separated.
- Below 0.3: Suggests that the clusters are poorly defined and may indicate that the chosen number of clusters is inappropriate for the data.

From the guideline above, we think that the K-means clustering done on the dataset did not work out very well. We can conclude that the clusters which were formed would be poorly defined and not well separated at all.

To further improve the on this result, we proceeded with an improved clustering which treats the data with dimensionality reduction first.

K-means clustering with prior dimensionality reduction

From the course of study, we learnt that dimensionality reduction methods can be used to remove irrelevant features so that the model can be more focused on components that capture the primary variance in the data. By reducing the noise, clusters that are generated can become more well-defined; model performs better.

For the case of this project and the topic at hand, the dimensionality reduction employed is Truncated SVD, which is first performing SVD on the dataset followed by removing axes that have low variance but still preserving enough axes so that the sum of squared variances is kept above 80%. This method is employed in the code snippet as below:

```
# use svd dimensionality reduction and choose the number of dimension that
keeps 80% of the total energy
from sklearn.decomposition import TruncatedSVD

# using the same scaled_hobbies from the previous code
# Calculate Truncated SVD
svd = TruncatedSVD(n_components=scaled_hobbies.shape[1] - 1) #n_components
must be < n_features
svd.fit(scaled_hobbies)

# Explained variance ratio
explained_variance_ratio = svd.explained_variance_ratio_

# Cumulative explained variance
cumulative_variance = np.cumsum(explained_variance_ratio)

# Find the number of components to retain 80% of variance
n_components = np.argmax(cumulative_variance >= 0.8) + 1

# Apply dimensionality reduction
svd = TruncatedSVD(n_components=n_components)
reduced_hobbies = svd.fit_transform(scaled_hobbies)
reduced_hobbies.shape
```

The output of reduced_hobbies.shape is **(1010,17)**. This means that the reduced hobbies data frame has 17 features which contribute to >= 80% of the variances. This reduced hobbies data frame is then subjected to the same elbow method and silhouette analysis.

The results of this are as follows: {Cluster #: # of respondents} = {0: 252, 1: 260, 2: 313, 3:185}. The silhouette score for this clustering is **0.093** which is a slight improvement from **0.074** in the previous. This is still a very low score and there's still no well-defined clusters.

We noticed that applying machine learning algorithms have not been very successful in helping us find out meaningful clusters of young people. We think that there might be 2 reasons for this:

1. The list of hobbies and interests given in the survey are too numerous and diverse
2. Young people sometimes have very diverse hobbies that might be unique combinations of their own.

Therefore, with this constraint, we would find it challenging to draw form meaningful clusters of people who might like the same range of hobbies. To mitigate both these reasons mentioned above, we propose to do 1 thing:

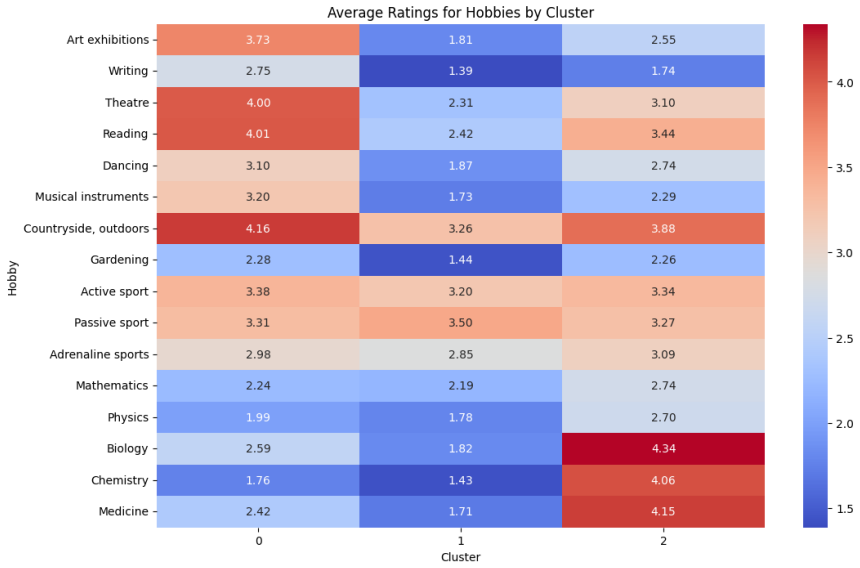
- Narrow down the list of hobbies and only select them from the areas of Arts & Culture, Outdoor Activities and Mathematics & Sciences

The implementation is as shown in the code snippet:

```
# Select only the specified hobbies
selected_hobbies = hobbies_df[['Art exhibitions', 'Writing', 'Theatre',
'Reading', 'Dancing', 'Musical instruments', 'Countryside, outdoors',
'Gardening', 'Active sport'
, 'Passive sport', 'Adrenaline sports', 'Mathematics', 'Physics',
'Biology', 'Chemistry', 'Medicine']]
```

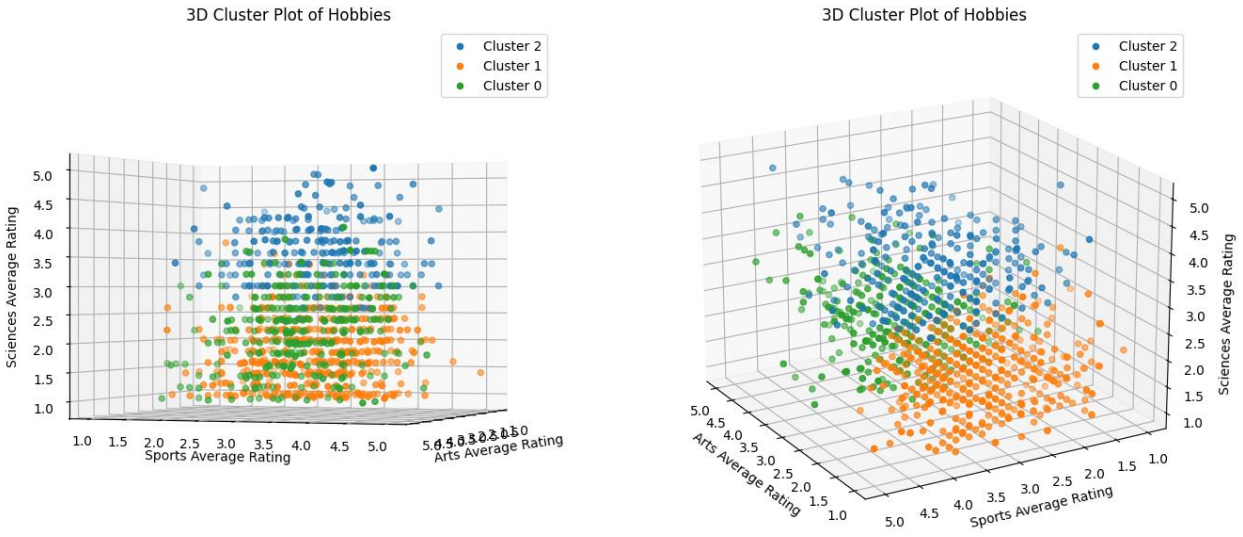
The results are as follows: {Cluster #: # of respondents} = {0: 314, 1: 454, 2: 242}. The silhouette score for this clustering is **0.126** which is a slight improvement from **0.093** in the previous.

While this is not a high score, we will explore further through visualization whether we can make sense of the clustering. We do so by visualizing the average ratings of the hobbies of each cluster and visualizing it on a heatmap:



From the heatmap visualization above, we can tell that each of the clusters have some defining characteristics which allows us to tell them apart from their average ratings. For example, cluster 0 are mostly active but do not really like partaking in Arts & Culture and Mathematics & Sciences.

We explore this further with a 3D scatter plot by grouping the selected hobbies into hobby groups which we previously used to select them. Each young person's rating for the 3 hobby groups will be the average of the constituent. The results are as below:



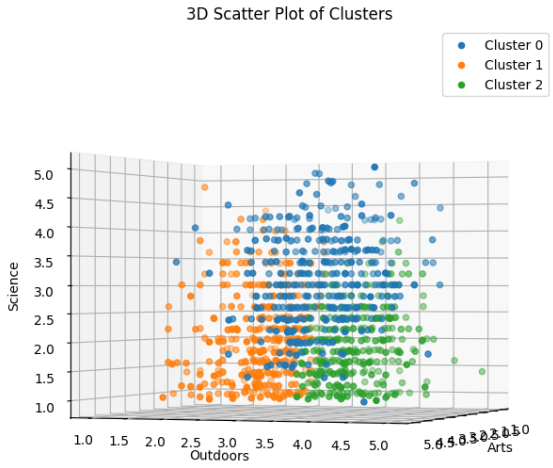
From the 2 scatterplots above, we observe that the 3 clusters generated might not be distinctly apart, but we have some indications on where their interests lie.

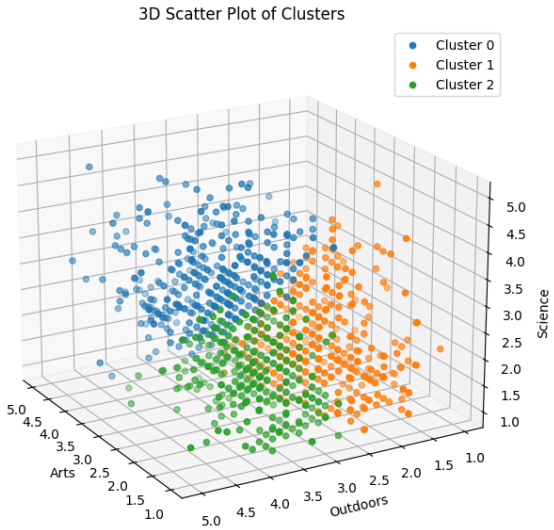
We make one final change to the data which is to group the selected hobbies into the 3 hobby groups before running the K-means algorithm. Code snippet as such:

```
# Calculate the means for the three groups of columns
arts_mean = hobbies_type3[['Art exhibitions', 'Writing', 'Theatre',
'Reading', 'Dancing', 'Musical instruments']].mean(axis=1)
outdoors_mean = hobbies_type3[['Countryside, outdoors', 'Gardening',
'Active sport', 'Passive sport', 'Adrenaline sports']].mean(axis=1)
science_mean = hobbies_type3[['Mathematics', 'Physics', 'Biology',
'Chemistry', 'Medicine']].mean(axis=1)

# Create a new DataFrame with the means
result_df = pd.DataFrame({'Arts': arts_mean,'Outdoors': outdoors_mean,
                          'Science': science_mean})

# Scale the data
scaler = StandardScaler()
scaled_result = scaler.fit_transform(result_df)
```

The results are as follows: {Cluster #: # of respondents} = {0: 311, 1: 358, 2: 311}. The silhouette score for this clustering is **0.245** which is a substantial improvement from **0.126** before grouping.

The visualization of this data can be displayed as a 3D scatterplot as well:





From these two scatterplots, we can tell the 3 clusters apart more clearly compared to the previous scatterplots above.

Future Extensions: Dimensionality Reduction and Clustering Algorithms

Through the above code, we have employed 2 different types of algorithms:

1. For point clustering algorithm, we have employed K Means clustering.
2. For dimensionality reduction algorithm, we have employed truncated SVD.

For the purposes of exploring clustering beyond the Jupyter notebook, there are 2 more methods which I could employ to derive different clustering results. They are:

1. PCA - Principal Component Analysis: which is another dimensionality reduction technique which simplifies the complexity of high-dimensional data while retaining the most important features. It transforms the original variables into a new set of variables called principal components.
2. GMM - Gaussian Mixture Models: This model is probabilistic and assumes that the data is generated from a mixture of several Gaussian distributions. GMM can be used for clustering the young people using their hobbies data with the assumption they are Gaussian PDFs. GMMs allow for more flexibility than K-means allowing for each datapoint to potentially fall under multiple clusters.

Graphing and Network Analysis

Instead of diving deeper into either PCA or GMM from dimensionality reduction and clustering respectively, we will employ another type of algorithm from graphs and networks. Using this separate technique, we hope to further explore and understand the survey responses in other useful and meaningful ways.

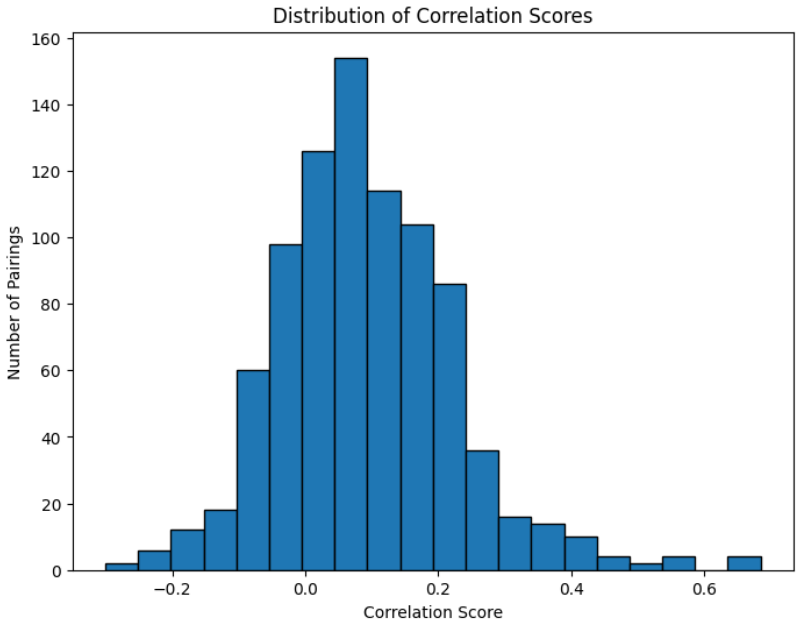
We hypothesize that the hobbies dataset can be explored further using graphing and network analysis algorithms. This would first require the conversion of the data which is collected on the Likert Scale to a graph data type. After which, we apply graph clustering/ community detection via the Louvain Method to group nodes on the graph to form meaningful clusters.

In the first step, we convert the dataset from ordinal data from the Likert Scale to correlation data between hobbies and identify high correlation scores between hobby pairings. We do so with the following steps:

1. Calculate the Spearman correlation matrix of the hobbies dataset
2. Exclude the diagonal entries in the matrix since those are correlation scores with the hobbies themselves
3. Plot the distribution of the correlation scores

```
# Calculate the Spearman correlation matrix
spearman_corr = hobbies.corr(method='spearman')

# Exclude the diagonal elements
spearman_corr =
spearman_corr.mask(np.equal(*np.indices(spearman_corr.shape)))

# Flatten the correlation matrix and remove NaNs
correlation_scores = spearman_corr.values.flatten()
correlation_scores = correlation_scores[~np.isnan(correlation_scores)]
```

After obtaining the correlation scores, we plot the distribution of the correlation scores as below:

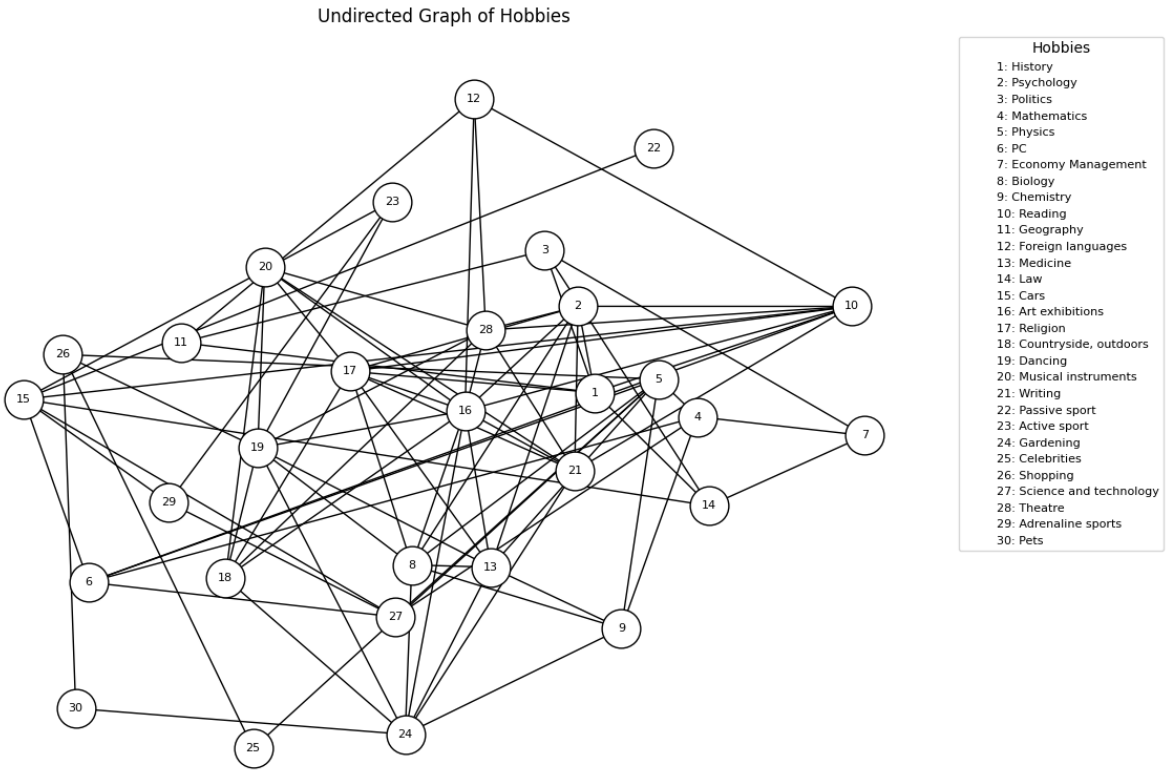


By observing the distribution above and knowing that the average correlation falls around **0.093**, we chose **0.2** as the "Threshold Correlation Score" for us to convert the Likert Scale ordinal data into a graph-like data.

We implemented the conversion of data type as such:

```
# Step 1: Calculate the Spearman correlation matrix
spearman_corr = hobbies.corr(method='spearman')

# Step 2: Create an undirected graph based on the Spearman correlations
threshold = 0.2  # Define a threshold for significant correlations
G = nx.Graph()

# Add edges for significant correlations
for i in range(len(spearman_corr)):
    for j in range(i + 1, len(spearman_corr)):
        if abs(spearman_corr.iloc[i, j]) > threshold:  # Check if the
correlation is significant
            G.add_edge(i + 1, j + 1, weight=spearman_corr.iloc[i,j])
#Use numbers as node identifiers

# Ensure all nodes are added to the graph
for i in range(1, len(spearman_corr) + 1):
    G.add_node(i)
```

The resulting graph looks as below:



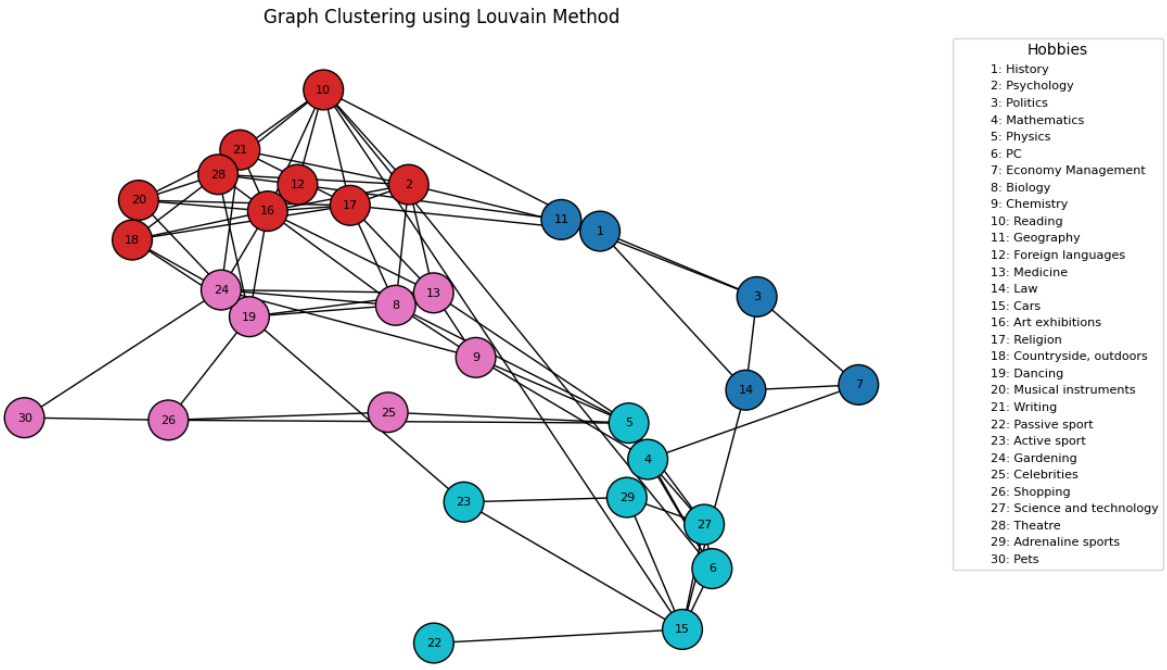
For the next step, as with our original intentions of finding hobby clusters so as to give suggestions to young people, we proceed to perform graph clustering. The method which we will employ is the Louvain Method which is briefly recapped in the steps below:

1. Each node starts as a community
2. The nodes are iteratively reassigned to communities based on modularity gain
3. When modularity gain cannot be increased further at the current level, each community is merged into a single super-node on its corresponding smaller graph
4. Steps 1 to 3 are repeated until maximum modularity is achieved

The implementation of the Louvain Method is direct in python as follows:

```
 # Compute the best partition using Louvain clustering
partition = community_louvain.best_partition(G)
```

Showing the results of the graph clustering on the graph below:



From the graph above, we observe that the Louvain Method has clustered the nodes on the graph into 4 different clusters. We first investigate the quantitative performance of the Louvain Method:

Modularity Score: **0.51**

Conductance of Communities:

- Community 1 - Members: [1, 3, 11, 14, 7], Conductance Score: **0.30**
- Community 2 - Members: [2, 10, 17, 16, 21, 28, 12, 18, 20], Conductance Score: **0.28**
- Community 3 - Members: [8, 13, 9, 25, 26, 19, 24, 30], Conductance Score: **0.43**
- Community 4 - Members: [4, 5, 6, 27, 15, 22, 23, 29], Conductance Score: **0.30**
- Average conductance: **0.33**

From the Modularity Score, which measures the density of edges within communities compared to what would be expected in a random graph, a general guideline is that scores above 0.3 indicate some structure within the graph while scores above 0.5 represent good community structure. With a modularity score of **0.51**, we can conclude that the clusters we have obtained are relatively well-defined.

However, with the caveat that high modularity doesn't always mean best community detection, we also use conductance of the communities to test. Conductance measures the

ratio of edges that leave a community relative to the internal edges. We gather from the conductance scores that Communities 1,2 and 4 are quite well-separated from the rest of the graph given that they have either 0.3 or lower conductance. We observe however that Community 3 has a slightly higher conductance value of **0.43**. We will investigate its nodes to understand the high conductance score better.

Community 3 consists of these members: [8, 13, 9, 25, 26, 19, 24, 30], which corresponds to these hobbies: [Biology, Medicine, Chemistry, Celebrities, Shopping, Dancing, Gardening, Pets]. From prior knowledge, we can tell that Nodes 8, 13 and 9 might not be too similar to the rest of the hobbies within the community. To that end, we tried to form a new community and have 5 communities in total by forming a 5^{th} community with just 8, 13 and 9:

Modularity Score: **0.50**

Conductance of Communities:

- Community 1 - Members: [1, 3, 11, 14, 7], Conductance Score: **0.30**
- Community 2 - Members: [2, 10, 17, 16, 21, 28, 12, 18, 20], Conductance Score: **0.28**
- Community 3 - Members: [25, 26, 19, 24, 30], Conductance Score: **0.6**
- Community 4 - Members: [4, 5, 6, 27, 15, 22, 23, 29], Conductance Score: **0.30**
- Community 5 - Members: [8, 13, 9], Conductance Score: **0.71**
- Average conductance: **0.44**

However, this does not improve the Modularity and Conductance but performs poorly.

Future Extensions: Graphing and Graph Clustering

While the above graphing and clustering methods arrived at a satisfactory set of clusters there are still a couple of ways in which the results can be improved. For this, we first have to look into the prior assumptions that were used when we approached the young people survey responses as a network. The assumptions that we took are:

1. We assumed that we could use correlations between hobbies to determine whether we should add an edge between nodes. We choose the edges based on a selected threshold for the correlation score.
2. Upon setting up the edges between nodes, we assumed that all the edges are the same weight

The above assumptions influence the graph and clustering in correspondingly:

1. While we used **0.2** as the correlation threshold criteria based on our observation of the correlation scores distribution, we understand that the choice of the threshold greatly influences the density/connectedness of the graph
2. Given that the edges were constructed out of correlation scores, we could possibly include the correlation value as an edge weight. Doing so will benefit the following analysis since the weights will preserve the strength of the relationship rather than treating all the weights as equal.

With these improvements in mind, we propose the following extensions that can be explored for this dataset pertaining to graphs and clustering:

1. We propose that the edges should also be accompanied by their associated correlation value between the paired nodes which shares the same edge
2. The edge weight can be accounted for when community detection/ clustering is done. This will help to reveal/conserve groups of related topics based on correlation.

We think that these adjustments made to the set up of the graph clustering algorithm will help us to optimize the community detection of the graph, i.e. the grouping of 8, 9 and 13 could very likely exist as a community (while maximizing modularity gain and minimizing community conductance) since their correlation scores with one another are very much higher compared to the scores between themselves and other hobbies.

Alternatively, there are also other graph clustering algorithms that can be employed on the graph itself. One that comes to mind is Approx Personalized PageRank which can be applied to the graph and finding clusters based on observing the local minima on a Conductance against Node Rank plot.

Summary

In this report, we have taken the responses of the young people survey and came with the following recommendations that young people can take reference to:

- Through dimensionality reduction and point clustering algorithm (K-means), we find out how to group young people who have commonalities amongst multiple hobby groups.
- Also, with the use of graphing and graph clustering algorithms, we can give suggestions to individual young persons on some of the hobbies which they might enjoy given the hobbies' membership in particular cluster groups (or we can refer to as hobby group)

We sincerely hope that the findings of this report can be useful for young people as they navigate the most exciting part of their lives, which is filled with transitions and novel experiences. We believe that the findings in our report are non-exhaustive and can be subjected to further extensions as mentioned above.