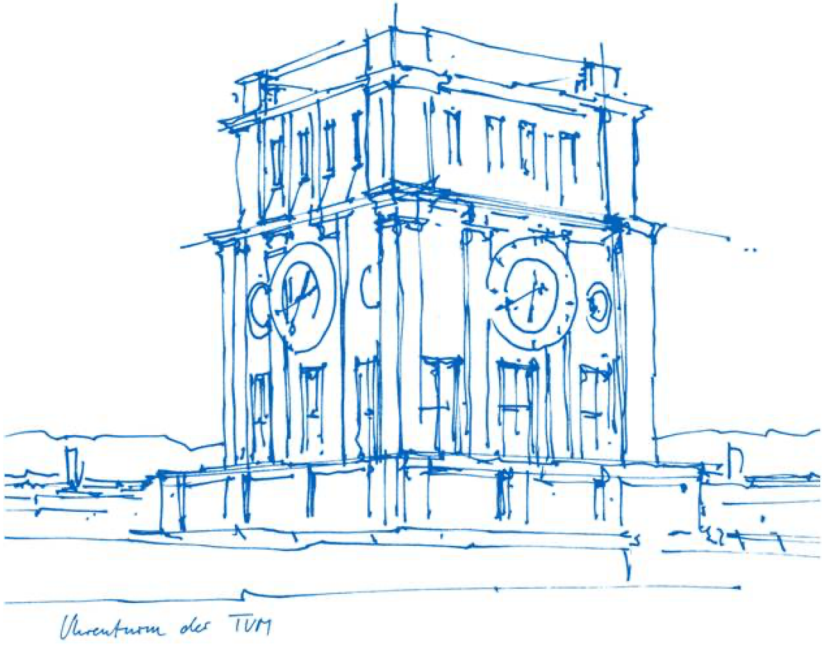
Exercises for
Social Gaming and
Social Computing
(IN2241 + IN0040) – Introduction to

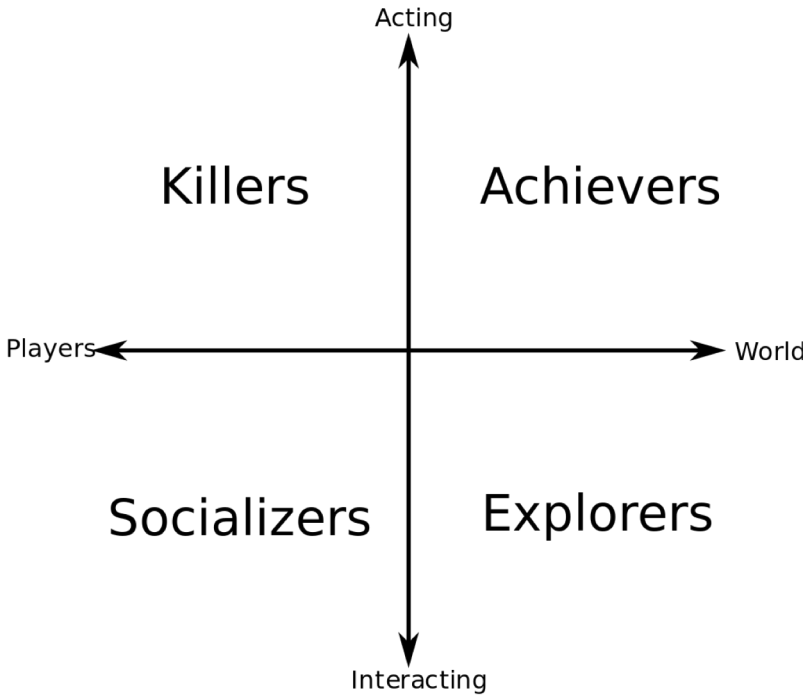
Exercise Sheet 5

Clustering with WoW



Exercise Sheet 5: Clustering with WoW

- goal: revise and compare K-Means clustering algorithm to Gaussian Mixture Models and DBSCAN
- process data gathered from the MMO-RPG *World of Warcraft*.
- cluster players into 4 groups (Radoff):
  - Socializers
  - Achievers
  - Explorers
  - Killers



The Data: wowah.csv

- wowah.csv - For every player:
  - ***char***: an ID for each character
  - ***level***: the level of the character
  - ***race***: the race of the character
  - ***charclass***: the class of the character
  - ***zone***: the area/zone the character was at when the data was gathered
  - ***guild***: the numeric ID of a guild
  - ***timestamp***: a timestamp of the moment the data was gathered.

K-Means

- K-Means: objective function (see lecture):

$$J(\mu) = \sum_{k=1}^{K} \sum_{\{n|x_n \in C_k\}} ||x_n - \mu_k||^2$$

where $\mu_k \in \mathbb{R}^d$ and $x_n \in \mathbb{R}^d$

Gaussian Mixture Models

- Gaussian Mixture Model:
  - Linear combination of Gaussians

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \quad \text{where} \quad \sum_{k=1}^{K} \pi_k = 1, \quad 0 \leqslant \pi_k \leqslant 1$$

parameters to be estimated

DBSCAN

- DBSCAN: rough idea (see slides)

iterate:
visit previously unseen pattern x:
if in ε-neighborhood {x'} of x: |{x'}| ≥ minPt then
start new cluster: include x and {x'} and those of their
ε-neighborhoods {x''} that are dense enough (|{x''}| ≥ minPt), etc.
else: x is noise

Tasks

Task 5.1: Preparation

Now that you are armed with all the knowledge needed, let us begin.

a) First, **read** the `wowah_data.csv` and `zones.csv` that you downloaded with this exercise into seperate dataframes.

We will gather all types of zones and assign them to one of the four following categories:

- **0** (social): cities and arenas
- **1** (zones): all zones, transists, seas and event areas
- **2** (dungeons): all dungeons
- **3** (battlegrounds): all PvP zones

b) After all zone types where assigned their new values, replace the zones from `data` with the new labels in the `zone_list`.

Hint: Use the `DataFrame.replace` function from pandas for b)

Now we will create a new dataframe for the playtimes by zones.

The total playing time per character is calculated as follows:

- We assume each player only plays until midnight
- If the difference between two timestamps is greater than one hour, we assume that this did not occure during the same session

c) Compute the difference between two following timestamps and save the playtime in the dictionary `zones_playtime` and add it to `total_playtime`.

After calculating the playingtime by zones for each player we want to add the relative playingtimes by zone to our new dataframe. This is necessary, since different players might have invested quite a different amount of time into this game.

Tasks (cont.)

Task 5.2: Clustering

a) k-Means Cluster the dataset with the k-Means algorithm and print out the centroids.

Hint: For the clustering we will use the k-means algorithm provided by the scikit-learn library. Import the algorithm and use the `fit()` function to let the algorithm do its work. Remember to set the amount of clusters to 4.

b) Gaussian Mixture Model Cluster the dataset using `GaussianMixture` from `sklearn.mixture`. Again we want to an end result of 4 clusters.

Hint: In order to retrieve the labels, we have to use the `predict()` function. Save the labels at `gmm_labels`.

c) DBSCAN Cluster the dataset using `DBSCAN`.

Before we can cluster with DBSCAN, we have to identify the correct parameters for the algorithm. We can find the optimal epsilon value by plotting the distances between the datapoints and taking the value at the point of highest curvature [2].

Tasks (cont.)

Task 5.3: Analysis

In the next part we are going to analyze our clustering results. For this we will calculate the means of each cluster in our Gaussian Mixture Model and map the clusters using t-SNE.

a) Means Before we visualize our data, we want to check if our clusters make sense. In order to do so, we can compute the means of our gmm clusters. This will also help us to identify the groups of player types later on. Write down your observations and try to map the clusters to the player types.

b) t-SNE ¶

Since we used 4 features for clustering our data lies in a 4-dimensional space. Visualizing and interpreting a 4-dimensional space can be tricky, therefore, we introduce an algorithm called [t-SNE] http://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf) [3] that can transform a high dimensional dataset into a 2 dimensional plot. For more information you can check out the linked paper. For a simple but intuitive explanation have a look at this video [4].

1. Run the given code to generate a t-SNE graph. Look at the plot and choose **one** cluster from the k-means visualization which you want to analyze. Can you tell what kind of player type the cluster represents in Bartle's model? Can you explain the meaning of the distance between the clusters?

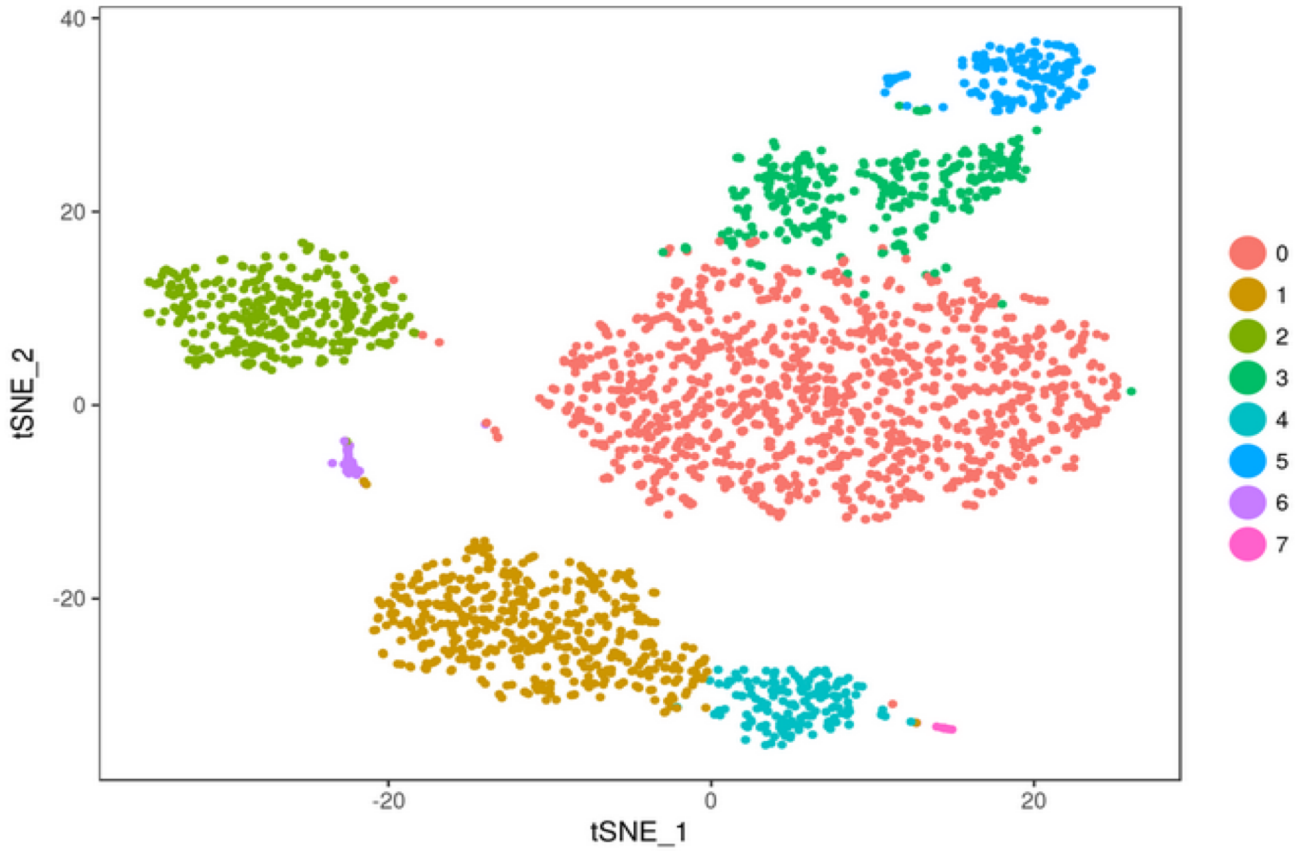
2. Compare the different clusterings and try to find similarities and differences. What could be reasons for these?

Hint: You can see the assigned clusters for each player with the list `kmeans.labels_` and `dbscan.lbabels_`

Note: If you get the impression that the clustering is not very accurate do not feel discouraged as the data set does not contain enough information about the other activities of the players besides ship killing.

t-SNE[5]

- designed to plot multidimensional data in a 2 dimensional grid
- helps to visualize data and identify clusters and similarity between clusters

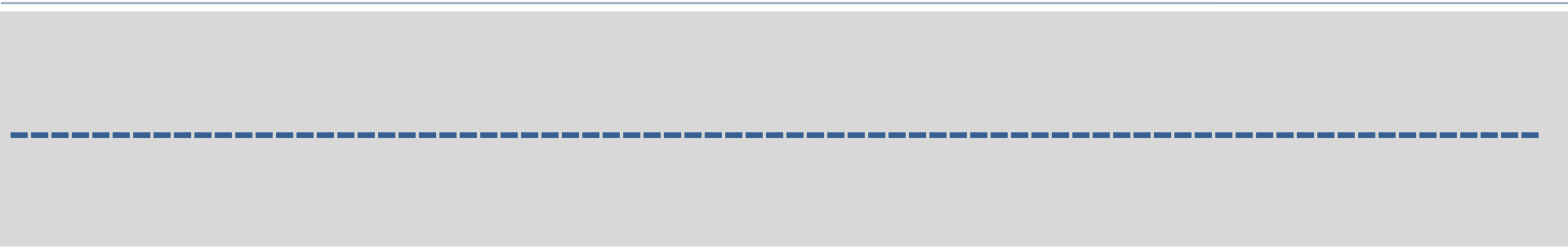


Tips and tricks

- before you start coding, familiarize yourself with the data
- you don't have to code the clustering algorithms yourself, make use of all tools at your disposal

Submitting your solution

- work by expanding the .ipynb iPython notebook for the exercise that you downloaded from Moodle
- save your expanded .ipynb iPython notebook in your working directory
- submit your .ipynb iPython notebook via Moodle (nothing else)
- remember: working in groups is not permitted. Each student must submit their own .ipynb notebook!
- we check for plagiarism. Each detected case will be graded with 5.0 for the whole exercise
- deadline: check Moodle



Citations

1. https://www.kaggle.com/mylesoneill/warcraft-avatar-history?select=wowah_data.csv
2. https://towardsdatascience.com/machine-learning-clustering-dbscan-determine-the-optimal-value-for-epsilon-eps-python-example-3100091cfbc
3. http://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf
4. https://www.youtube.com/watch?v=NEaUSP4YerM
5. Laurens van der Maaten, Geoffrey Hinton: Visualizing Data using t-SNE, 2008 (PDF)