Assignment 10

Statistical Machine Learning

Moritz Haas / Prof. Ulrike von Luxburg

Summer term 2022 — due on **July 4th at 12:00**

Exercise 1 (Spectral graph theory 2+1+2+1=6 points)
In the following we consider the symmetric normalized graph Laplacian $L = D^{-1/2}(D - A)D^{-1/2}$, where A denotes the adjacency matrix of an undirected and unweighted graph and D the graph's degree matrix, for various special graphs.

(a) Let G be an undirected, unweighted, and connected graph. Let $0 = \lambda_1 < \lambda_2 \leq \ldots \leq \lambda_n$ denote the increasingly sorted eigenvalues of L. Prove that

$$\lambda_n \leq 2.$$

Hint: For all $a, b \in \mathbb{R}$, it holds that $(a + b)^2 \leq 2(a^2 + b^2)$. Why? And how does this help you?

(b) Let G be the complete graph on n vertices, that is, any two vertices are connected. Show that its Laplacian has eigenvalues 0 with multiplicity one and $n/(n-1)$ with multiplicity $n - 1$.

(c) Let G be the complete bipartite graph on $n+m$ vertices, that is, every vertex $v \in \{v_1, \ldots, v_n\}$ is connected with every vertex $\tilde{v} \in \{v_{n+1}, \ldots, v_{n+m}\}$, but no two vertices in $\{v_1, \ldots, v_n\}$ and $\{v_{n+1}, \ldots, v_{n+m}\}$ are connected. Show that its Laplacian has eigenvalues 0 with multiplicity one, 1 with multiplicity $m + n - 2$, and 2 with multiplicity one.

(d) Let G be the star on n vertices, that is, v_1 is connected with every vertex $v \in \{v_2, \ldots, v_n\}$ but no two vertices in $\{v_2, \ldots, v_n\}$ are connected. Show that its Laplacian has eigenvalues 0 with multiplicity one, 1 with multiplicity $n - 2$, and 2 with multiplicity one.

Exercise 2 (Similarity graphs, 2+2+1=5 points)

Many machine learning algorithms build on graphs. In this exercise we explore similarity graphs, a simple method to transform data into graph representation.

(a) Implement the function `plot_similarities(X, sigma)`. It takes a data matrix X as input and computes the paired similarities according to the Gaussian kernel with parameter σ

$$s(x_i, x_j) = \exp(-||x_i - x_j||^2/(2\sigma^2)).$$

The similarities are then visualized in a heat map. In the interactive simulation, play with the kernel parameter σ on the *original two moons* and the *noisy two moons* dataset. Which values of `sigma` would you choose, and why? What is the role of noise, how does it influence your choice of `sigma`?

(b) Now implement the function `plot_graph(X, k, dim, mutual)`, which uses `kneighbors_graph` of `sklearn.neighbors` to create a k-nearest neighbor graph and and plot its adjacency matrix. Only the first `dim` dimensions of the input data `X` should be used to create the graph. Based on the parameter `mutual`, the graph should be either symmetric or mutual.

In the interactive simulation, play with the parameters `k` and `dim` on the *unbalanced Gaussian* data set. For $d = 2$, what is the smallest `k` for which the graph consists of a single connected component? How do both graphs look like for small and high values of `k`? Now start to increase the dimension. What happens for $d = 200$? Do you have a (theoretical) explanation for this effect?

(c) Now implement the function `plot_degree(X, k)`, which again creates a k-nearest neighbor graph and plots the vertices in the graph with different colors, depending on their degree.

In the interactive simulation, play with the parameter `k` and see how the vertex degrees change. For a suitable choice of `k`, there is a relation between vertex degrees in the k-nearest neighbor graph and the probability distribution from which the data is drawn. Can you guess what it is?

Exercise 3 (Spectral Clustering, 1+1+2+2+1=7 points)

(a) In X you find a $n \times 2$ matrix with $n = 400$ samples in $\mathbb{R}^2$ from three clusters. Apply the built-in function `sklearn.cluster.KMeans` with $k = 3$ (applies k-means++ by default), plot the data and color the points according to their assigned cluster. Mark the cluster centers.

Repeat this for $k = 2$ and $k = 4$.

(b) Implement a function `kneighbors_graph(X, num_neighbors)` which takes as input the data points (X_i) with $i = 1, \ldots, n$ and parameter `num_neighbors`. Return the $n \times n$ weight matrix of the corresponding symmetric kNN-graph W where $W_{i,j} = 1$ if X_i is among the `num_neighbors` nearest neighbors of X_j or vice versa. Otherwise set $W_{i,j} = 0$.

Plot the kNN graph for the smallest `num_neighbors` such that the graph is connected.

Hint: You can use the function `interact` *from Exercise 2 to create interactive plots and play around with the parameters. Plotting kNN graphs with many edges can take a few seconds.*

(c) Implement a function `spectral_cluster(W, k, normalize, regularization = 0)` which takes as input a kNN-graph, a parameter k for the number of clusters (not for the kNN-graph), a boolean value indicating whether to use the normalized or unnormalized Laplacian and an optional regularization parameter (see lecture slides 798ff). The function should perform spectral clustering, run `KMeans` on the spectral embeddings and return the centers, the clustering as well as the embedded data points as a $n \times k$ matrix Y.

In the next task you can use the built-in function `sklearn.cluster.SpectralCluster` if you are not confident in your own implementation.

(d) Apply unnormalized spectral clustering to the data set from task (a). Choose a reasonable parameter `num_neighbors` for the kNN-graph and try to detect three clusters.

Make 3 plots ($j = 1, 2, 3$), where data point i has a color representing its embedding value Y_{ij} with respect to the j-th eigenvector of the employed Laplacian (where eigenvalues are sorted increasingly).

For the resulting clustering make the same plot as in (a) to visualize the cluster membership of each point. Visualize the spectral embedding in a 3-d scatter plot.

Repeat this with normalized spectral clustering.

(e) Now load the file `sbm_adjacency.npy`, which contains a graph adjacency matrix without additional information. The graph was generated by a random graph model called 'Stochastic Block Model' and consists of several communities/clusters/blocks. Choose a reasonable number of clusters k using the spectral embedding and give an estimate of the number of nodes in each cluster.

Exercise 4 (Design your own exam questions, 3 points)
In this exercise, everybody is supposed to come up with suggestions for three exam questions. This is a good way to recap/understand the concepts discussed so far.

Put yourself in our place! We do not want to ask stupid questions. We would like to ask "nice questions". In general, written exams contain three types of questions:

- Questions that are just about **reproducing** knowledge. These kind of questions are pointless and we don't ask questions of this kind in the exam.

- Questions for testing whether the person **understands** the concepts and can apply them to simple situations.
- Questions that require to **transfer** knowledge to new situations.

Your task is now to design exam questions along with their solutions of the two last-mentioned types. The questions can be about any topic that the lecture has covered so far. Enter your questions in the LaTex file `my_exam_questions.tex` that we provided and send it to your tutors.

After the class we will put all your questions online. At the end of the course, these questions can help everybody to prepare for the exam!