Künstliche Intelligenz und Data Science WS 2023/2024

Lab 4: Graph Algorithms

Due: January 28, 2024, 11.59 CET

Dr. Vincenzo Perri
vincenzo.perri@uni-wuerzburg.de
Prof. Dr. Ingo Scholtes

Machine Learning for Network Analysis

Social network data, typically represented as graphs with nodes symbolizing individuals and edges capturing relationships, lends itself well to exploration through network science methodologies. Analyzing the structural properties of the network allows for the revelation of key organizational features and underlying dynamics, such as the identification of cohesive groups. Integrating machine learning techniques further enriches network analysis by introducing a predictive and analytical dimension. For instance, supervised learning models can be trained on labeled data to discern patterns, facilitating the identification of nodes with specific characteristics or roles within the social network.

This assignment guides you through an exploration of social network data incorporating methodologies from both network science and machine learning. The network structure is obtained from the aggregation of face-to-face interactions between 92 company employees recorded in an office building over a period of ten days [1]. The node labels represent the departments of the employees. The node features are synthetically generated to complement the assignment.

Background on dimensionality reduction

In many real-world scenarios, datasets contain numerous features, some of which may be redundant or irrelevant to the given prediction problem. Dimensionality reduction is a crucial technique in data science that seeks to identify and preserve the most informative aspects of the data. The process helps visualize data and improves the ability of machine learning algorithms to generalize their results to new data. In fact, retaining a smaller number of essential features reduces the possibility that a model fits the noise in the data and, therefore, the chance of overfitting.

This assignment requires you to use the following two dimensionality reduction methods.

Principal Component Analysis [2] (PCA) is a widely used method for Euclidean datasets. PCA transforms the original features into a new set of uncorrelated variables called principal components. These components capture the maximum variance in the data, allowing for a reduced-dimensional representation that maintains as much information as possible.

Laplacian Eigenmaps [3] (LE) is a commonly employed dimensionality reduction technique for network data. Specifically, Laplacian Eigenmaps utilize the graph Laplacian matrix to give the nodes a lower-dimensional vector representation that preserves the graph structure. This low-dimensional representation highlights latent structures and relationships between nodes, such as densely connected communities. Notice that this algorithm builds on ideas closely related to that of Spectral Clustering (see Lecture 18 of AKIDS2)

While the assignment requires you to use dimensionality reduction techniques, you are not expected to implement them from scratch and can treat them as black boxes. For PCA, we recommend using the implementation available in scikit-learn. For Laplacian Eigenmaps, we make available the `LaplacianEigenmaps` function contained in `LAB_04.ipynb`.

Assignment

1. **Data Loading and Exploratory Analysis:**
   - Load the network data, including the edgelist, node labels, and node features. Load the network as undirected, i.e., the existence of edge (u, v) implies that of (v, u).
     Suggested packages for analyzing network data:
     - `pathpy`
     - `networkx`
   - Use the `decomposition` module from `sklearn` to Apply PCA on node features and reduce their dimension to two. Create a scatter plot with nodes colored according to their labels. What does the plot tell us about the features' ability to classify nodes according to their labels?
2. **Logistic Regression Classification:**
   - Use the `train_test_split` function from `sklearn`'s `model_selection` module to create an 80/20 train/test split of the data. Use the two sets to train a `LogisticRegression` model and evaluate its performance by computing the accuracy score and by computing and plotting the confusion matrix.
   - Compare the confusion matrix to the scatter plot of PCA-projected node features from point 1. Do they provide a conflicting or a similar perspective?

3. **Node Labels and Network connectivity** This set of tasks uses probability to explore the connection between the patterns in the network topology and node labels.

   - Compute the Probability Mass Function (PMF) of label occurrence P(l) and visualize it using a bar plot.
   - Use the edges and the node labels to compute the empirical PMF of inter-label connections,$P(l, l')$, for each label couple (l, l'). In other words, compute the fraction of times edges connect nodes in with labels (l, l').
   - Compute the PMFs of inter-label connections using $P(l)$ and assuming independent connections between nodes with different labels.
   - Use a bar plot to compare the two PMFs. What connections are largely more common in the empirical connections? What does this tell us about the way connections form in this system?

4. **Laplacian Eigenmaps & Bayes Rule:** This set of tasks uses conditional probabilities and the Bayes rule to explore the relation between the vector representation returned by Laplacian Eigenmaps and node labels.

   - Use the provided `compute_laplacian_embedding` function to compute a $d = 5$ dimensional vector representation of the network's nodes.
   - Bin each eigenvector's values into 50 equal-width bins and plot the resulting five PMFs.
     Hint: use the `hist` function from `matplotlib.pyplot`
   - Compute m_i, the median value of each eigenvector $\mathbf{e}_i$, and create bar plots for $P(l|\mathbf{e}_{i,v} > m_e)$, i.e., for the probability of a node v having label l conditioned on its corresponding value in the i^{th} eigenvector being larger than the median value of the eigenvector m_i.
   - Use Bayes rule to invert the conditional probability and obtain $P(\mathbf{e}_{i,v} > m_e|l)$, i.e., the probability that node's v value at the i^{th} eigenvector is larger than the median conditioned on having label l. Visualize the resulting probabilities using bar plots.

5. **Optimal Dimension for Classification and Feature Combination:**

   - Search for the optimal dimension for classification (80% training 20% test) of both node features and Laplacian Eigenmaps. For the provided node features, fix the dimensionality with PCA. For the features from the Laplacian, fix the number of eigenvectors. Consider increasing dimensionalities from $d = 1$ to the maximum number of dimensions (number of nodes minus one for the Laplacian and the total number of features for the provided data) with steps $\Delta d = 5$. Perform $n = 10$ classification experiments for each dimensionality.

Suggestion: Fix an array of $n = 10$ randomly sampled random states to give in input to the `train_test_split` function. This will ensure that your splits remain the same for both types of features and for all dimensions.

- Perform the classification using both types of features together. Compare the confusion matrix of this classification with that of the classifications obtained using only one type of feature. How did combining the two types of features affect the result?

Provided Files

- `workplace2013.edgelist`
- `labels.csv`
- `node_features.csv`
- `LAB_04.ipynb` (containing LaplacianEigenmaps function)

Submission

Submit a zip file with Python files and provided data.

References

[1] Génois, Mathieu and Vestergaard, Christian L and Fournet, Julie and Panisson, André and Bonmarin, Isabelle and Barrat, Alain, *Data on face-to-face contacts in an office building suggest a low-cost vaccination strategy based on community linkers*, Network Science, 2015, Cambridge University Press

[2] `https://en.wikipedia.org/wiki/Principal_component_analysis`

[3] Belkin, Mikhail and Niyogi, Partha *Laplacian eigenmaps for dimensionality reduction and data representation*, Neural computation MIT Press