CS-E4650 Methods of Data mining
Home assignments 3

Deadline Sun 1.11. 2020 23:59
Choose only 4 tasks from the following 5. Each task has maximum 25 points (total 100 p).

1. In this task, you should experiment with the Gephi tool `https://gephi.org/users/download/` and analyze a social network among school children. Install Gephi, open it and install one extra plugin ('tools → plugins'), Newman-Girvan clustering. You can also install all updates ("check for updates").

   Data `schoolclass5day1.csv` contains a matrix presentation of the interaction graph among all class 5 pupils during one day. The edge weight reflects the strength of interaction (total duration). In file `schoolclass5meta.txt` you can find some extra information on pupils: the class (5A or 5B) and gender (F or M).

   Analyze `schoolclass5day1.csv` with Gephi. When you open data, set graph type as 'undirected'. You can find instructions in Gephi guide `https://gephi.org/users/quick-start/` but the appearance has changed a bit in the newest version, so check also `https://github.com/gephi/gephi/issues/1258`. The functions are available under 'statistics' and 'data table' shows values of nodes for all calculated measures. After running a function, you can also visualize results under 'appearance → ranking'.

   a) Identify most central and influential nodes with different measures (node degree and weighed degree, closeness centrality and betweenness centrality). You can look at 5–8 most central nodes (set a suitable threshold for each measure). What do these measures tell about nodes? Do they rank the same nodes as most central?

   b) Identify communities with two alternative methods, Modularity and Girvan-Newman clustering, and compare results (similarities and differences). Does the background information (class and gender) explain communities?

   c) Present the communities visually. Use distinct colours to show nodes, add labels, hide low weight edges to simplify the graph, move nodes so that communities become separated. Then take snapshots of both Modularity and Girvan-Newman results.

d) In this subtask, use the communities detected by the Modularity function. Test hiding low weight edges with different thresholds and analyze links inside and between communities: Which communities have strongest interconnections? What are bridge nodes that combine two communities (end points of strong links between communities)? Are these the same as central nodes? Or what is the role of central nodes in communities?

2. Consider a graph $G = (V, E)$. Given a subset of vertices $S \subseteq V$, we propose the following definition of density:

$$\frac{\sum_{i \in S} \sum_{j \in S} w(i,j)}{|S|},$$

where $w(i,j)$ is the weight of the edge between vertices i and j, if it exists, and 0 otherwise, and $|S|$ is the cardinality of the set S. Note that in unweighted graphs, $w(i,j) \in \{0,1\}$.

a) Propose a greedy algorithm to find a subgraph with good density according to this definition.

b) Evaluate the results on the *dolphins.txt* data set. In this file, the first line contains the number of vertices. Each subsequent line indicates the two endpoints of an **undirected** edge.

3. Let us consider the task of analyzing and mining subgraph patterns. Figure 1 shows an example of six molecular graph structures. The node labels correspond to atoms (carbon, oxygen or nitrogen)[1]. Three of the molecules, graphs G_1 (serotonin), G_3 (dopamine) and G_6 (melatonin) belong to class M (monoamines), while G_2 (acetaminophen = paracetamol), G_4 (ibuprofen) and G_6 (caffeine) belong to class $\neg M$.

a) Identify maximum common subgraphs (MCG) and calculate MCG-based distances between class M molecules and from class M molecules their nearest neighbours. It suffices to identify two nearest neighbours to each class M graph, unless there are many equally close neighbours.

Compare two MCG-based distances: Udist and Mdist (Equations 17.2 and 17.3 in Aggarwal's book). Which one separates class M molecules better from other molecules?

[1]For simplicity hydrogen atoms and double bonds between atoms are not presented.

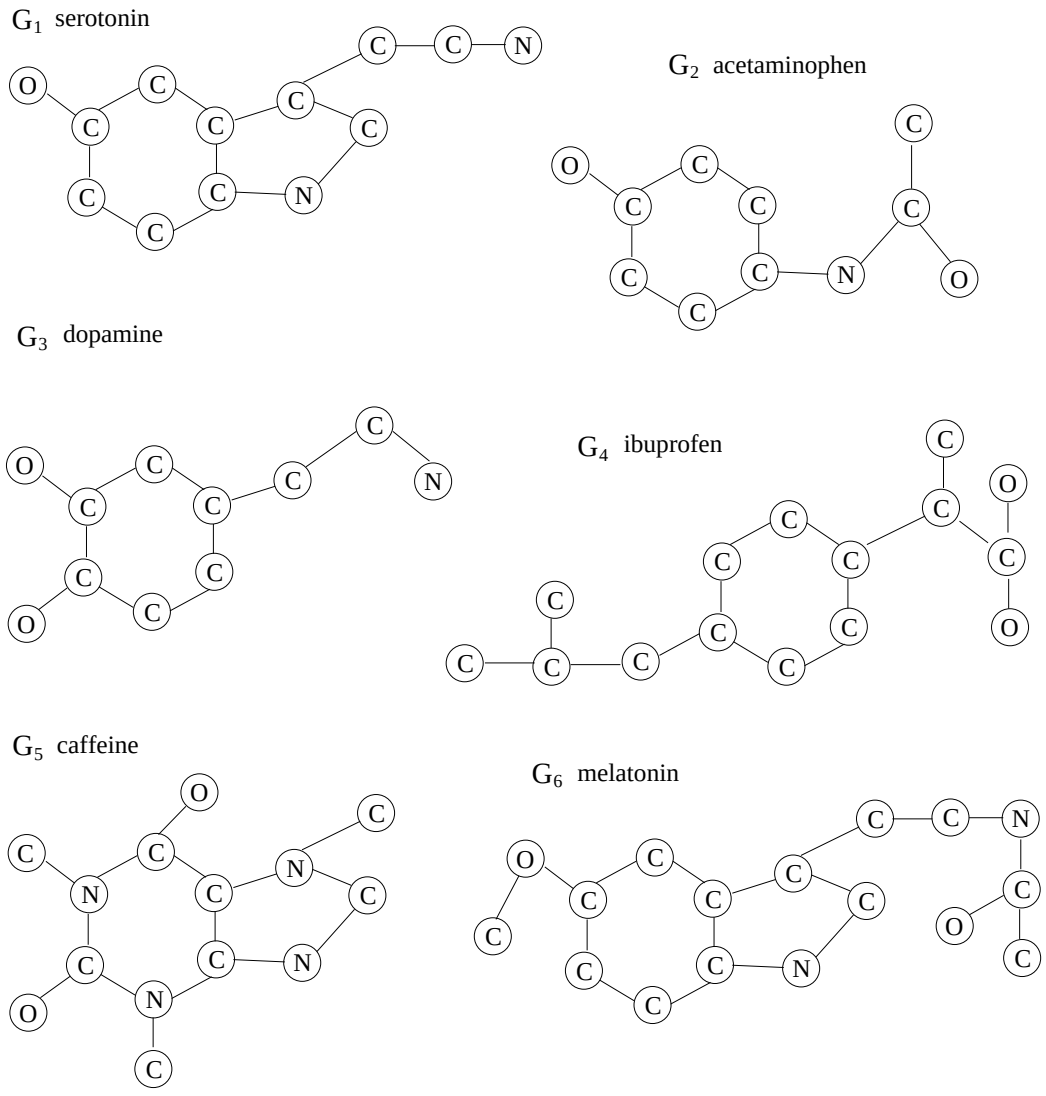

Figure 1: Six graphs corresponding to molecule structures.

b) Identify maximum common subgraph G for all class M molecules. You can draw it by hand if you want. How well does it predict class M? I.e., calculate precision ("confidence") of graph–class rules $G \rightarrow M$ and $\neg G \rightarrow \neg M$, where rule condition G means that subgraph G occurs in the graph and $\neg G$ means that it doesn't.

c) In real databases, molecules have often many characteristic attributes like toxic, drug, amino acid, etc. An important task is to identify subgraphs that are statistically associated with different

attributes. Show that statistical dependency between occurrence of subgraph G and binary attribute A is not a monotonic property, i.e., there could be positive dependency rule $G \rightarrow A$, even if all subgraphs G' of G were independent or negatively associated with A.

d) Describe how you can search for statistically significant associations describing positive dependencies between subgraphs and class attributes using GraphApriori. Is this a good approach? Consider such aspects as the effect of minimum frequency threshold, problems of missed or redundant patterns and computational efficiency.

4. Implement a recommendation system utilizing SimRank or personalized PageRank algorithms. Given a user_id, the recommendation system is required to output the top K most recommended items that the user has not yet bought and the predicted rating.

   Implement a two-phased approach, where you first search for the K (e.g., 20) most similar users with personalized PageRank or SimRank and then make the recommendations using ratings in that peer group.

   Use the data jester-800-10.csv to construct the recommendation system and make then recommendations to users in test-800-10.csv. The dataset contains ratings of 10 jokes by 800 users. The rating is 1 if the users give a positive feedback to the joke and 0 otherwise.

   You can also test the predictions with yourself (rate only some jokes and let the system recommend you other jokes).

   Return the source code and a report that documents how you implemented the recommendation system, how you tested its predictions and what are your conclusions on its performance.

5. This task continues exercise session 3 task 4 on classifying movie reviews as positive or negative. The starting point is a simple classifier presented in section 1.3 ("Document classification") of chapter 6 of the book Natural Language Processing with Python `http://www.nltk.org/book/ch06.html`).

   a) Modify the code such that the document features are based on the frequencies of words in positive and negative reviews. Compare the accuracy of the two representations.

      You may want to preprocess the documents, e.g. remove stopwords:

```
from nltk.corpus import stopwords
print(stopwords.words('english'))
```

b) Word features can be very useful for performing document classification, since the words that appear in a document give a strong indication about what its semantic content is. However, many words occur very infrequently, and some of the most informative words in a document may never have occurred in our training data. One solution is to make use of a lexicon, e.g., WordNet lexicon (`http://www.nltk.org/howto/wordnet.html`), which describes how different words relate to one another.

In WordNet, information about words is represented using synsets. A synset of a word is a set of synonyms that share a common meaning.

```
from nltk.corpus import wordnet as wn
```

Find all synsets of word 'horror':
`wn.synsets('horror')`
WordNet returns:
[`Synset('horror.n.01'), Synset('horror.n.02'), Synset('repugnance.n.01')`]

Word 'horror' has three synsets that represent different meanings of the word. For this assignment, we can assume that the first alternative in the list is the most useful.

Find the first synset (object):
`horror = wn.synset('horror.n.01')`

A hypernym is a word with a broader meaning, for instance 'fear' is a hypernym of 'horror'.

Find the hypernyms of the 'horror' object:
`horror.hypernyms()`
returns [`Synset('fear.n.01')`]

Using WordNet lexicon (`http://www.nltk.org/howto/wordnet.html`), augment the movie review document to use hypernyms of the words that appear in a document. You can replace a word in features with its hypernym or add hypernyms as extra terms, or

try both. You can also choose to replace just part of the words, e.g. low-frequency words.

In your report, describe the features used in your experiments and compare the accuracy of the augmented classifier(s) to the classifier presented in the book. Discuss the results. Return the code.