Michele Sprocatti, ID 2121719
Riccardo Zuech, ID 2128872
Paul Cariou, ID 2133045

Learning From Networks Project Proposal

Exploring the power of the clustering coefficient as a feature of popularity in a co-purchase network of books

Motivation:

Given a co-purchase network for products in the book category of amazon.com, the clustering coefficient gives us a strong metric of correlation between books bought together, allowing us to explore the following questions:

1. Given the popularity of genres, according to the clustering coefficient of the books, how do they distribute and how do they compare with the popularity results obtained according to the sales rank?
2. Given the following joint definitions of popularity:

	High cc	**Low cc**
High sales rank	Popular	Niche-Popular
Low sales rank	“Topic-Specific”/Must have	Unpopular

   How does the situation change considering these four categories?

Data:

The dataset of product co-purchased from amazon.com is available at Amazon product co-purchasing network metadata. In particular, the dataset will be parsed in order to contain only products (nodes) from the book category.
For each product we have important information available:

- “Similar”: tells us the IDs of other books that are usually bought together with the one considered.
- “Salesrank”: provides us the “popularity” of books according to Amazon.
- “Categories”: provides us with the tree of subgenres for each book.
- Other meta information, such as ratings and reviews, is provided and may prove useful.

The graph, comprising 548552 nodes (393561 of them are books) was collected during the summer of 2006.

Methods:

- *Problem:* compute the local clustering coefficient for each node.
    - *Algorithm:* we will attempt to apply the exact local clustering coefficient algorithm to the dataset. If this proves infeasible, we will revert to the approximate version discussed in class.
- *Problem:* define “popularity” according to clustering coefficient and salesrank:
    - *Approach:* test “above average” values as a popularity definition and tune it according to the results and the underlying distributions.
- *Problem:* compute the popularity of genres at different sub-genres levels, given that a book may be part of many different genres.
    - *Approach:* build a tree-like data structure where each node is a genre and its children the sub-genres; given a popular book, starting from the leafs of the tree (most specific sub-genres of the book) travel upwards toward the root updating an internal counter in the nodes proportional to the appearance of that sub-genre in the book’s classification.

Intended Experiments:

- *Libraries:* We decided to use the NetworkX library given its straightforward use on Google Colab, in contrast to other solutions.
- *Machine for experiments:* Google Colab with default configurations:
    - Intel(R) Xeon(R) CPU @ 2.20GHz
    - 13 GB of RAM
- *Experiments:* We will attempt to apply the exact local clustering coefficient algorithm to the graph described above. If this proves infeasible, we will revert to the approximate version discussed in class.
  In particular, we want to:
    1. Explore the popularity of genres, at different sub-genre levels, according to the clustering coefficient and compare it with the results obtained from the popularity according to the sales rank.
    2. Given the following joint definition of a book by its local cc and sales rank:

	High cc	**Low cc**
High sales rank	Popular	Niche-Popular
Low sales rank	“Topic-Specific”/Must have	Unpopular

Investigate the distribution of genres, at different sub-genre levels, among these four categories of books.

Specifically, the points above will be implemented through the use of the tree of genres data-structure described in the methods section.

Bibliography:

1. *Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014*

2. *Jure Leskovec and Rok Sosiˇc. Snap: A general-purpose network analysis and graph-mining library. ACM Transactions on Intelligent Systems and Technology (TIST), 8(1):1, 2016*