Assignment 2: Predicting missing links in citation networks

MSc Data Science & Business Analytics – ESSEC Business School | CentraleSupélec

Team: **Five Sigmas**

Rafaëlle Aygalenq	Sarah Lina Hammoutene	Dóra Linda Kocsis	Ouiza Ouahes	Noémie Quéré
B00724587	B00712035	B00714326	B00722231	B00719656

INTRODUCTION

The following report details the feature engineering and link prediction mechanisms implemented on a citation network of research articles. The network is represented as a graph G=(V, E), where the nodes are scientific articles and a directed edge between two nodes u and v, indicates that paper u cites paper v. The following information is also at disposal for each node: title of the paper, publication year, author names, short abstract.

The goal is to reconstruct the initial network using graph-theoretical and textual features. Solutions were evaluated using the F1 score and accuracy. During the competition our team reached a score of: 97.85% and a ranking of 01/19.

SECTION 1: FEATURE ENGINEERING

1.1 DATA & PRE-DEFINED FEATURES

Before feature engineering, we decided to opt for a different data representation from the original one. Given our baseline assumption, that similarity breeds connection [1], we converted the data into pairs of nodes and assigned each with a unique ID. This way, we achieve an easier and computationally faster and cheaper way to extract new features for the node pairs.

The following baseline features were transformed and implemented on the dataset:

- `number of overlapping words in paper titles`
- `number of common authors`
- `temporal distance between the papers`

We then split our graph in two and used 95% for the graph features and the remaining 5% to validate the models. Therefore, we primarily have 2 types of feature: text-based and graph-based. The following section will provide further details on these.

1.2 TEXT-BASED FEATURES

Beyond exploiting the features using the node metadata, we implemented the `number of overlapping words` *for abstracts* as well. We then applied text processing techniques and graph theory to obtain the features explained in further detail in the following sections.

FEATURES USING TF-IDF

In order to be able to compare the similarity between two nodes, we first computed the term-frequency matrices of the titles and abstracts of the nodes contained in each node pair respectively. Afterwards, the cosine distance was calculated to obtain the similarity features between the node pairs:

- `Cosine distance between two Tf-idf titles`
- `Cosine distance between two Tf-idf abstracts`

Tf-idf

Term frequency-inverse document frequency (tf-idf) is often used in information retrieval and text mining. This statistical measure corresponds to the importance a word has in a document or a collection of documents.

Tf-idf allows to compute the relevance of a word within the corpus as its value increases with the frequency of a word in a document but it is offset with the frequency of the word in the corpus. Thus, for a word *i* in document *j*:

$$w_{i,j} = tf_{i,j} + log(\frac{N}{df_i})$$

$tf_{i,j}$ = number of occurrences of i in j
df_i = rarity of i across a set of abstracts j
N = total number of abstracts

Cosine distance

In order to be able to compare the similarity between two nodes the cosine distance of their tf-idf matrices has to be calculated.

Cosine similarity is then obtained by taking the dot product of the two matrices and dividing that by the product of their norms, yielding the cosine of the angle between them:

$$cos(\theta) = cos(x_i, x_j) = \frac{x_1 \cdot x_2}{\|x_1\| \|x_2\|}$$

The cosine similarity of two documents will range from 0 to 1, as the tf-idf weights are always positive. Furthermore, the angle between two term frequency vectors (i.e.: the cosine similarity) cannot be greater than 90° for the same reason.

FEATURES USING LATENT SEMANTIC ANALYSIS

Latent Semantic Analysis (LSA) is a technique for creating a vector representation of a document, more precisely for analyzing relationships between a set of documents and the

terms they contain. In practice, we use SVD to perform dimensionality reduction on the tf-idf vectors. Then, we can again apply the cosine similarity procedures that we defined earlier. Thanks to the rank reduction of the original matrix, we obtain an approximation of the document-term matrix, that gives us a new representation of each document in our corpus.
We added 2 features using LSA:

- `the cosine distance between two LSA abstracts`
- `the cosine distance between two LSA titles`

1.3 GRAPH-BASED FEATURES

For these features, we built a new graph from the 95% of the original data and saved in a txt file. The following additional features were considered:

NUMBER OF CITATIONS BETWEEN AUTHORS
Papers with similar topics of interest should have a lot of common citations, as they are treating the same topics, and thus quoting each other often. Hence, the graph must be composed of sub-graphs by topic of interest. Thus, we computed and added the number of citations between authors to our features.

NUMBER OF COMMON IN & OUT NEIGHBORS BETWEEN ARTICLES
If two nodes have many common connections, they must be connected to each other. Therefore, from our graph, we computed the `common in neighbors` (predecessors) and `common out neighbors` (successors).

JACCARD SIMILARITY IN & OUT
Using the above-mentioned features, we were able to add the Jaccard similarity for both in and out neighbors. This feature is essentially a similarity metric, measuring the probability, that both x and y have a feature f, for a randomly selected feature f that either x or y has. In this scenario, features are neighbors, so we computed the metric as the quotient of the number of common neighbors over the number of union of neighbors. This statistic is used to measure the similarity between finite sample sets.

SHORTEST PATH
One of the most direct metric for quantifying how close two nodes are is the shortest path distance from x to y. This metric was also calculated and added as a feature.

PAGERANK
PageRank computes a ranking of the nodes in the graph G based on the structure of the incoming links. Rooted PageRank can be defined as terminal node of a random walk starting from x. From x, it moves to a random neighbor with probability (1 – α), where α is a damped pre-defined parameter. Then, with probability α it returns to x. Thus, we get c(x, y) as the stationary probability of y in a rooted PageRank. We added the rooted PageRank as a new feature for our model [2][3].

We implemented this feature using the NetworkX PageRank function with an alpha or damping parameter of 0.7.

NODE2VEC
By definition[1] the node2vec framework learns low-dimensional representations for nodes in a graph by optimizing a neighborhood preserving object. It provides a way of balancing the exploration-exploitation tradeoff.

Here, we used the code, `node2vec` - included as an appendix to the report - to generate the embedding of the nodes. Once we obtained our embedding, we computed the similarity between node source and destination in the 5% that was initially kept for validation.

SECTION 2: MODEL COMPARISON

Given that we face a binary classification problem where the goal is to predict the existence of an edge between two nodes, we have to explore the capabilities of classification algorithms. We chose to implement the following models: **Logistic Regression (LR), Naive Bayes (NB), Support Vector Machines (SVM), Random Forest (RF), Gradient Boosting (GBM), XGBoost (XGB) and Neural Network (NNET).** For the sake of simplicity, we omit from describing these algorithms more in depths. Essentially, we obtained five intermediary results discussed in the following section.

BASIC FEATURES ONLY

	LR	NB	SVM	RF	GBM	XGB	NNET
F1	.708	.362	.797	.753	.802	.799	.807
ACC	.659	.550	.749	.715	.756	.753	.754

Features considered here were: `number of overlapping words in paper titles and abstracts, number of common authors, temporal distance between the papers.` Here, neural network and gradient boosting provided the highest F1 scores and accuracy.

[1] Source: http://snap.stanford.edu/node2vec/

ADDING TFIDF FEATURES

	LR	NB	SVM	RF	GBM	XGB	NNET
F1	.814	.739	.800	.845	.865	.864	.863
ACC	.805	.765	.754	.836	.854	.853	.846

Tf-idf features significantly improved both our F1-scores and accuracy, with Gradient Boosting providing the highest performance.

ADDING LSA FEATURES

	LR	NB	SVM	RF	GBM	XGB	NNET
F1	.853	.788	.845	.871	.887	.888	.884
ACC	.844	.799	.825	.863	.878	.879	.875

There is only a slight improvement after adding the LSA features, with XGBoost providing the highest performance on the validation set.

ADDING GRAPH FEATURES

	LR	NB	SVM	RF	GBM	XGB	NNET
F1	.943	.916	.938	.969	.972	.973	.959
ACC	.940	.913	.933	.966	.970	.971	.957

After adding, the number of citations between authors, the common in and out neighbors, the Jaccard similarity features, shortest path and PageRank, we achieved results over 90% for each algorithm, still with XGBoost giving the highest F1-score for the validation set.

ADDING NODE2VEC

To finally compare all the models presented before, we implemented a Monte Carlo cross-validation. 10 different train and test samples had been randomly created and all models had been computed. The accuracy scores for each method and each test sample were computed and plotted in the boxplot below (Figure 2). The figure indicates that the best performing models are: Random Forest, XGBoost and Gradient Boosting.

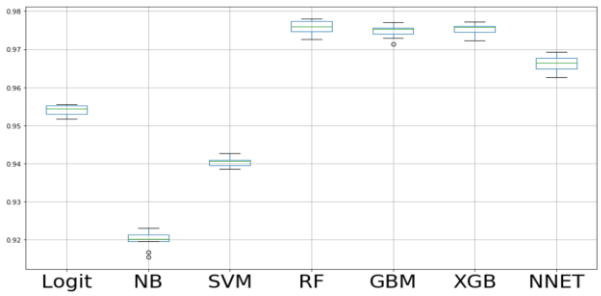

Fig.1 Final Model Comparison

Since Random Forest gives the best results among all models and is the easiest one to optimize among ensemble algorithms, we decided to choose this one as our final model. Using cross-validation technique we obtain the following optimized parameters:

- `n_estimators`: 500
- `max_feature`: auto
- `min_samples_leaf`: 3
- `min_samples_split`: 7
- `max_depth`: 50

Finally, to have an idea about how our final model is working, we plotted Fig 2, depicting feature importance of each feature for the Random Forest algorithm.

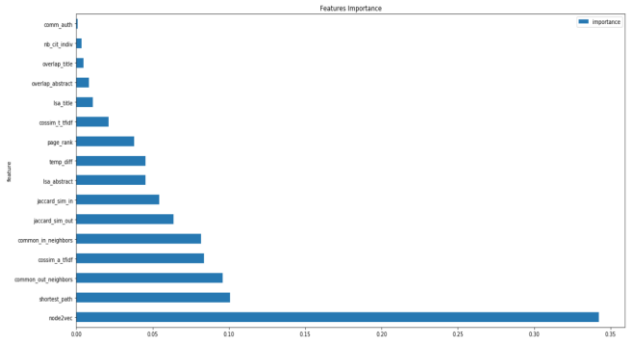

Fig.2 Feature Importance

As the figure shows, node2vec followed by the shortest path, common in & out neighbors and cosine distance between two Tf-idf abstracts are the top 5 features of our model.

During the competition our team reached a score of: 97.85% and a ranking of 01/19.

REFERENCES

[1] Ciotti et al. (2016) Homophily and missing links in citation networks. EPJ Data Science 2016 5:7

[2] William Cukierski, Benjamin Hamner, Bo Yang (2011) Graph-based Features for Supervised Link Prediction. Proceedings of International Joint Conference on Neural Networks, San Jose, California, USA

[3] David Liben-Nowell and Jon Kleinberg (2003) "The link prediction problem for social networks," in Proceedings of the twelfth international conference on Information and knowledge management, New York, NY, USA, 2003, CIKM '03, pp. 556–559, ACM.