1 Technical Report: Random Forest for Link Prediction

Link prediction is a fundamental task in network analysis that involves predicting potential links between nodes in a given network. Whether it's identifying future friendships in a social network or foreseeing potential interactions in biological systems, link prediction provides insights into the dynamics and structure of complex networks.

In this lab report, we explore the application of Random Forest classifiers in the context of link prediction. The goal in this technical report, is to investigate the effectiveness of this algorithm in predicting potential links within a network. In the essay following this technical report, the ethical implications of using such an algorithm will be investigated and several solutions will be proposed to mitigate some of the negative effects that may arise from deploying Random Forest algorithms for link prediction tasks.

The edges for the training graph is stored in `edges_train.edgelist`. Besides the edgelist, there is a complementary dataset called `attributes.csv`. The attributes label the nodes in the edgelist to one of five classes.

1.1 Feature Engineering

In this section, the use of features for both training and inference are elaborated. The edgelist is denoted as E.

Node attributes Each node has one of five attributes, denoted by a letter (e.g. "g" or "c"). For every two nodes $i, j \in E$ that make up a positive or negative edge, the node-attribute was one-hot encoded. This accounts for the first ten features.

Node degree As with the node-attributes, the degree of each node i and j in an edge is used. Degree is a measure of node importance. Nodes with higher degrees might be more likely to form connections. To increase diversity, the sum of the degrees of nodes i and j has been included as well as the absolute difference, resulting in four different node degree features. Feature diversity was further increased by applying MinMax-scaling to each of the four features, resulting in a total of eight node degree features.

Preferential attachment Preferential attachment suggests that nodes with higher degrees are more likely to form links with new nodes. This score reflects this principle.

Common neighbours Common neighbours indicate nodes that share connections. More common neighbours can imply a higher chance of forming a link.

Jaccard Coefficient The Jaccard coefficient measures similarity based on common neighbours. Higher Jaccard coefficient indicates that two nodes are likely to have a similar neighbourhood.

Adamic-Adar index The Adamic-Adar index is another measure of similarity based on common neighbours but it assigns lower weights to common neighbours with higher degree. It can capture the potential for link formation.

Resource allocation index Resource allocation index measures similarity based on shared neighbours, giving more weight to neighbours with lower degree. Higher index values can imply a higher likelihood of forming a link.

One can reason from the description of the selected features that many are quite similar. They are, however, complementary since each of then relies on another part of the existing neighbourhood patterns.

1.2 Data Split

Since we have prior knowledge that each test-set has an equal number of positive and negative edges, we have decided to use this as a heuristic to better tune the model. Therefore, the dataset was constructed by computing the edge features for *all* existing or positive edges (n=4368) and half the number of n (m=2184) of non-existing or negative edges. This was done to slightly bias the model in favour of predicting positive edges, to ensure the number of positive predictions is roughly equal to the number of negative predictions. The data was split into a training-, validation- and test set of 60%, 20% and 20% of data respectively.

1.3 Model Selection

For this link-prediction task, several models were considered. We selected Logistic Regression (a linear model), Decision Tree Classifier (a non-parametric model) and Random Forest Classifier as well as a GradientBoosting Classifier (both ensemble methods) to ensure there is a reasonable variety of models. Multinomial Naïve Bayes, was not considered since it is mainly designed to handle discrete non-negative values. Gaussian Naïve Bayes performed poorly at this task (likely due to its simplicity) and was therefore left out. The models were tuned for hyper-parameters using grid-search [1], they have been laid out in Table 1. One thing to note is that decision trees and their ensemble counterparts, are prone to over-fitting. One way of mitigating it, would be to prune the trees by setting a maximum depth. The Gradient Boosting algorithm required too many estimators, of too large depth to be considered reasonable. The Random Forest, being simpler than Gradient Boosting, but richer in complexity than a single Decision Tree is therefore the selected model.

Table 1: Hyperparameters for different models.

Model	**Hyperparameters**
Logistic Regression	C: 3 regularisation: L1
Decision Tree	max_depth: 7 max_features: sqrt
Random Forest	n_estimators: 55 max_depth: 25
GradientBoosting	n_estimators: 210 learning_rate: 0.5 max_depth: 50 sub_sample: 0.7

1.4 Model training & validation

To combat over-fitting, the models were validated using 10-fold cross-validation. The models trained using 10-fold cross-validation were significantly more robust than those trained on a 5-fold cross-validation. The final validation results are shown in Table 2.

Table 2: Validation results (mean and std.dev) of a 10-fold cross-validation (n=1966)

Model	**Precision**	**Recall**	F_1**-Score**	**Accuracy**
Logistic Regression	0.897 ± 0.026	0.706 ± 0.033	0.790 ± 0.024	0.748 ± 0.026
Decision Tree	0.830 ± 0.054	0.835 ± 0.030	0.840 ± 0.031	0.778 ± 0.048
Random Forest	0.885 ± 0.033	0.941 ± 0.022	0.913 ± 0.020	0.876 ± 0.031
GradientBoost	0.834 ± 0.043	0.907 ± 0.035	0.878 ± 0.019	0.818 ± 0.035

1.5 Final results

The predictive models were evaluated using the test-set which yielded the results shown in Table 3 and the ROC-curves for Decision Tree, Random Forest and Gradient Boosting are plotted in Figure 1. The results indicate that Random Forest performs the best for this link prediction task.

Table 3: Selected models evaluation on test-set (n=1966)

Model	**Precision**	**Recall**	**F_1-Score**	**Accuracy**
Logistic Regression	0.904	0.703	0.791	0.748
DecisionTree	0.861	0.875	0.868	0.819
RandomForest	0.927	0.951	0.938	**0.915**
GradientBoosting	0.835	0.940	0.885	0.834

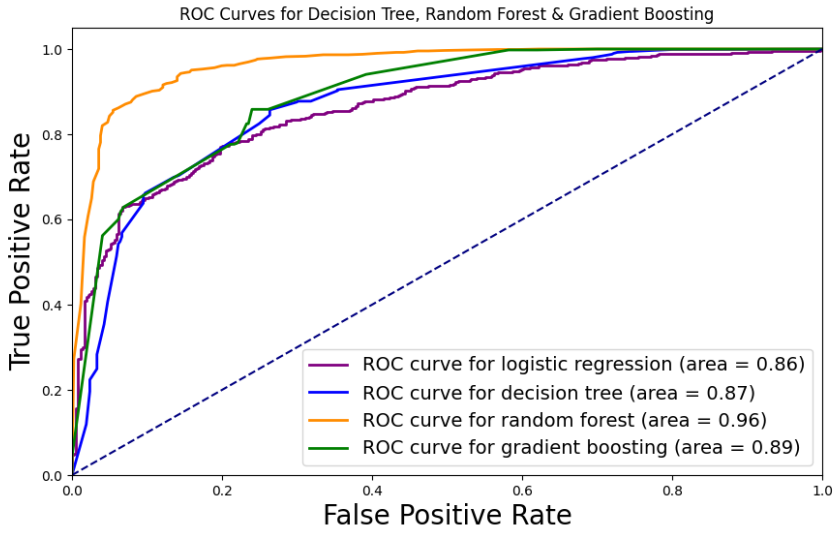

Fig. 1: ROC-curve for Decision Tree, Random Forest and Gradient Boosting

1.6 Conclusion & Technical Limitations

In this study, we utilised several machine learning models for the link-prediction task, including Logistic Regression, Decision Tree Classifier, Gradient Boosting, and Random Forest. However, each model presents its own set of technical challenges. In our experiments, Logistic Regression did not perform as well as the other selected models. Decision trees are prone to over-fitting [2], [3] and can exhibit high variance, potentially capturing noise in the data. On the other hand, Gradient Boosting and Random Forest are computationally intensive and due to the number of hyper-parameters, they require meticulous hyper-parameter tuning for optimal performance. To avoid over-fitting the Decision Tree, we used a 10-fold cross-validation instead of a 5-fold. Still, given that certain features rely heavily on the structural characteristics of the training-graph, link prediction for edges originating from a similar (yet still different graph) may not be as accurate as when the test-edges originate from the same graph as the training graph. The test-results obtained in this research are unlikely to generalise well given this constraint. It remains a matter of domain expertise to assess whether a training-graph is applicable to the area of deployment.