MACHINE LEARNING ON GRAPHS

Link prediction, Node classification, graph reconstruction, etc.

MACHINE LEARNING

- Wikipedia:
  - Machine learning(ML) involves computers discovering how they can perform tasks without being explicitly programmed to do so
- Subset of *artificial intelligence*
- Objective of machine learning: make a program learn automatically something about your data

MACHINE LEARNING

- Supervised Machine learning:
  - Train the program with examples (properties => associated value), the program can then predict the result given input properties
- Unsupervised Machine learning:
  - Given the data, the program should find by itself its rules/organization.
  - =>Most common example: clustering.
  - =>Community detection is unsupervised machine learning on graphs

MACHINE LEARNING

- Examples of supervised machine learning
  - Given properties of an apartment, predict its energy consumption
  - Given a picture, recognize objects in it
  - Given a student profile, predict its success
  - Given a criminal profile, predict its probability of recidivism
  - Given past values and collected news, predict market fluctuations
  - Given a patient profile, predict effect of a drug
  - Given a fingerprint/face, recognize the user
  - …

MACHINE LEARNING

- On graphs:
  - Link Prediction
  - Node Classification (feature prediction)
  - A few others(graph classification, graph reconstruction,…)

SUPERVISED MACHINE LEARNING 1: LINK PREDICTION

LINK PREDICTION

- Do you know why Facebook "People you may know" is so accurate?
- How youtube/Spotify/amazon recommend you the right item?
- =>Link prediction
  - More generally, recommendation, but link prediction is a popular way to do it

LINK PREDICTION

- Observed network: current state
- Link prediction: What edge
  - Might appear in the future (*future link* prediction)
  - Might have been missed (*missing link* prediction)

LINK PREDICTION

- Overview:
- Link prediction based on network structure:
  - Local: High clustering (friends of my friends will become my friends)
  - Global: Two unrelated hubs more likely to have links that unrelated small nodes
  - Meso-scale organisation: different edge probability for nodes in different communities/blocks
- Link prediction can also be based on node properties
  - e.g., age, revenue, gender, etc.
  - Combining with usual machine learning, outside of the scope of this course

FIRST APPROACH TO LINK PREDICTION:

HEURISTIC BASED

(HEURISTICS, NOT MACHINE LEARNING)

HEURISTICS

- Network science experts can design **heuristics** to predict where new edge might appear/be missing

- Principle: design a score based on network topology f(v1,v2) which, given two nodes, express their likeliness of being connected (if they aren't already)
  - Common neighbors
  - Jaccard coefficient
  - Hub promoted
  - Adamic Adar
  - Ressource allocation
  - Community based

Zhou, T., Lü, L., & Zhang, Y. C. (2009). Predicting missing links via local information. *The European Physical Journal B*, *71*(4), 623-630.

COMMON NEIGHBORS

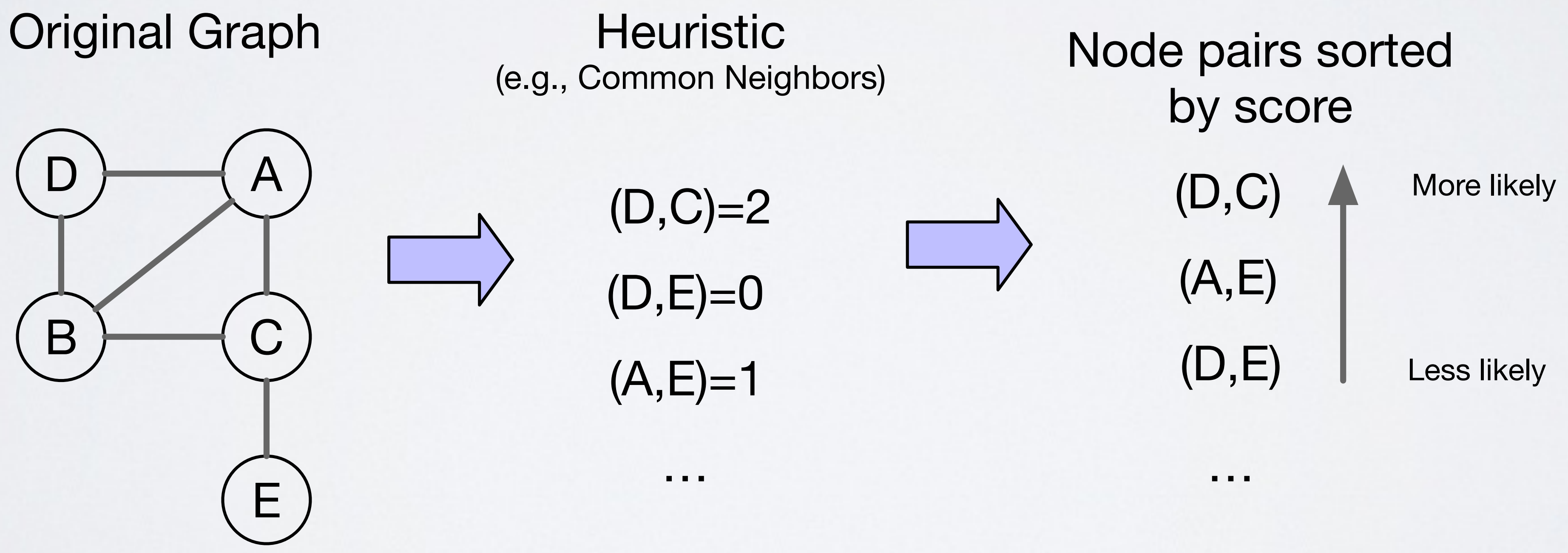
- "Friends of my friends are my friends"
- High clustering in most networks
- =>The more friends in common, the highest probability to become friends

$$\mathrm{CN}(x, y) = |\Gamma(x) \cap \Gamma(y)|$$

$\Gamma(x)$ = Neighbors of x

PREDICTION

- How to predict links based on Common Neighbors (CN)?



JACCARD COEFFICIENT

- Used in many applications:
  - Measure of similarity of sets of different sizes

$$\mathrm{JC}(x, y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|}$$

- Intuition:
  - Two people who know only the same 3 people
    - =>high probability
  - Two people who know 1000 people, only 3 in commons
    - =>Lower probability

HUB PROMOTED

- Intuition:
  - Normalized by min-number of neighbours
  - Variant: hub depressed (max instead of min)

  - Two stars have 10 common followers or I have ten friends following a star

$$\mathrm{HP}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{min(|\Gamma(x)|, |\Gamma(y)|)}$$

ADAMIC ADAR

- Intuition:
  - For previous scores: all common nodes are worth the same
  - For AA:
    - A common node with ONLY them in common is worth the most
    - A common node connected to everyone is worth the less
    - The higher the size of its neighborhood, the lesser its value

$$\mathrm{AA}(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{log|\Gamma(z)|}$$

RESSOURCE ALLOCATION

- Similar to Adamic Adam, penalize more higher degrees

$$\mathrm{RA}(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{|\Gamma(z)|}$$

$$\mathrm{AA}(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{log|\Gamma(z)|}$$

PREFERENTIAL ATTACHMENT

- Preferential attachment:
  - Every time a node join the network, it creates a link with nodes with probability proportional to their degrees
  - In fact, closer to the definition of the configuration model

- Score not based on common neighbors
  - =>Assign different scores to nodes at network distance >2

- Intuition: Two nodes with many neighbors more likely to have new ones than nodes with few neighbors

$$\mathrm{PA}(x,y) = |\Gamma(x)| \cdot |\Gamma(y)|$$

OTHER SCORES

Examples of other scores proposed

Sorenson Index

$$\mathrm{SI}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x)| + |\Gamma(y)|}$$

Salton Cosine Similarity

$$\mathrm{SC}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{\sqrt{|\Gamma(x)| \cdot |\Gamma(y)|}}$$

Hub Depressed

$$\mathrm{HD}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{max(|\Gamma(x)|, |\Gamma(y)|)}$$

Leicht-Holme-Nerman

$$\mathrm{LHN}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x)| \cdot |\Gamma(y)|}$$

COMMUNITY STRUCTURE

- General idea:
  - 1)Compute community structure on the whole graph
  - 2)Assign high score for 2 nodes in a same community, a low score otherwise
- How to choose the score?

COMMUNITY STRUCTURE

- For methods based on a quality function optimization (Modularity, Infomap's information compression, etc.)
  - Assign a score to each pair proportional to the change in quality function associated with adding an edge between them

- For instance, Louvain optimize Modularity.
  - Each edge added between communities:
    - Decrease in the Modularity
  - Edge added inside community:
    - Increase in Modularity, depends on properties of the community and nodes

Ghasemian, A., Hosseinmardi, H., & Clauset, A. (2019). Evaluating overfit and underfit in models of network community structure. *IEEE Transactions on Knowledge and Data Engineering*.

COMMUNITY STRUCTURE

- For Stochastic Block Models
- Reminder:
  - SBM assign each node to a community
  - For each pair of community, a probability of having an edge
- Probability of edge between pair:
  - Density between their respective communities
- If a Degree-Corrected SBM:
  - Probability also depends on degrees of nodes

OTHER SCORES

- Distance based:
  - Length of the shortest path
  - Probability to reach a node from another on a random-walk of distance *k*
    - See next class on embeddings
  - Number of paths of length *l* between the nodes

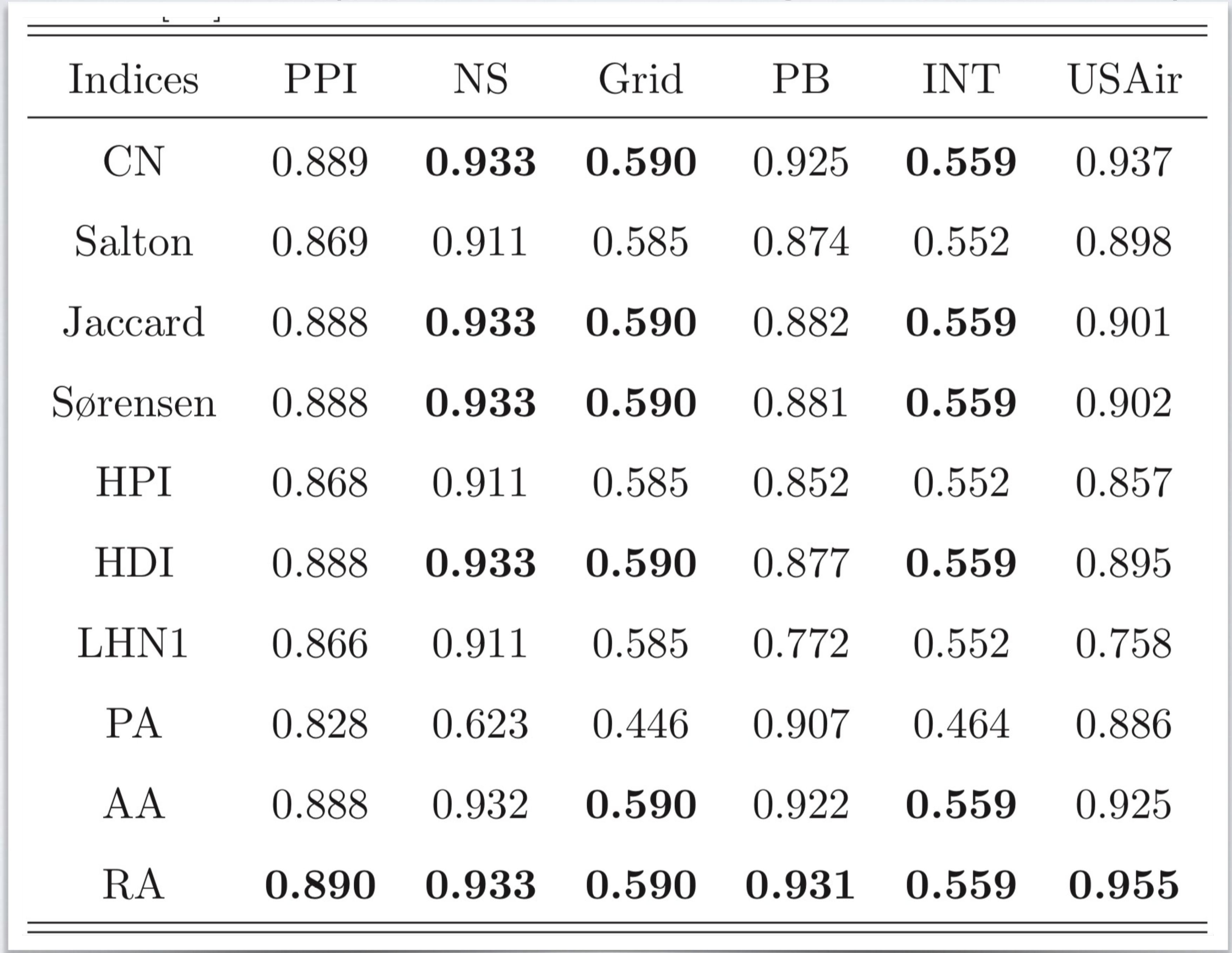
- Problem: computational complexity

COMPUTATIONAL COMPLEXITY

- To obtain the whole list of similarity scores: $\mathcal{O}(n^2)$
  - Intractable on large graphs
- In most cases, we care only about the top of the list
  - 2-hops-distances measures: Strongly reduce complexity
    - $\mathcal{O}(n\langle k\rangle^2)$
  - Preferential Attachment: compute among t top highest degree nodes
    - $\mathcal{O}(t^2)$

WHICH ONE IS BEST?

- Compute on many networks using AUC score (Explained later)

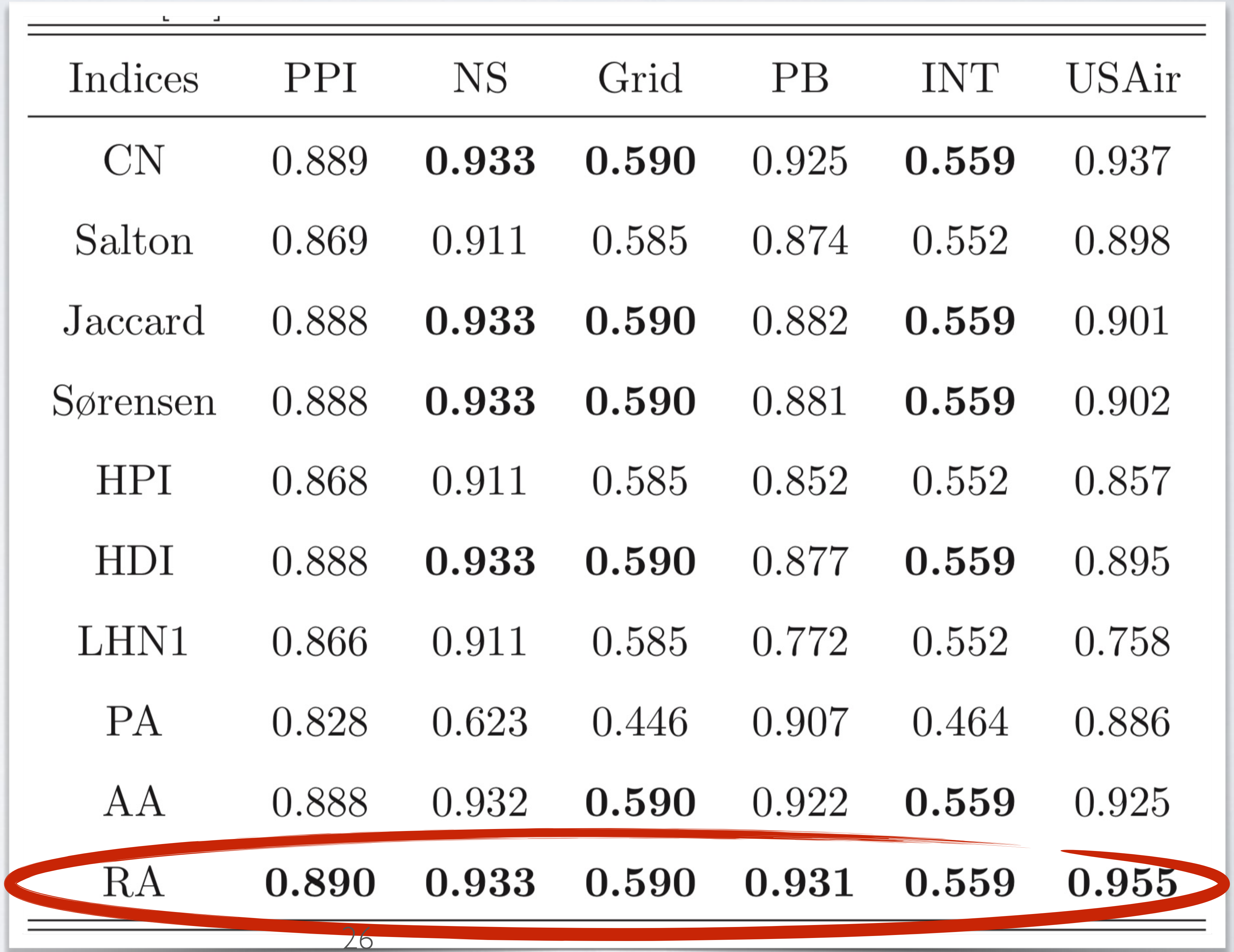


Indices	PPI	NS	Grid	PB	INT	USAir
CN	0.889	**0.933**	**0.590**	0.925	**0.559**	0.937
Salton	0.869	0.911	0.585	0.874	0.552	0.898
Jaccard	0.888	**0.933**	**0.590**	0.882	**0.559**	0.901
Sørensen	0.888	**0.933**	**0.590**	0.881	**0.559**	0.902
HPI	0.868	0.911	0.585	0.852	0.552	0.857
HDI	0.888	**0.933**	**0.590**	0.877	**0.559**	0.895
LHN1	0.866	0.911	0.585	0.772	0.552	0.758
PA	0.828	0.623	0.446	0.907	0.464	0.886
AA	0.888	0.932	**0.590**	0.922	**0.559**	0.925
RA	**0.890**	**0.933**	**0.590**	**0.931**	**0.559**	**0.955**

Zhou, T., Lü, L., & Zhang, Y. C. (2009). Predicting missing links via local information. *The European Physical Journal B*, *71*(4), 623-630.

WHICH ONE IS BEST?

- Compute on many networks using AUC score (Explained later)

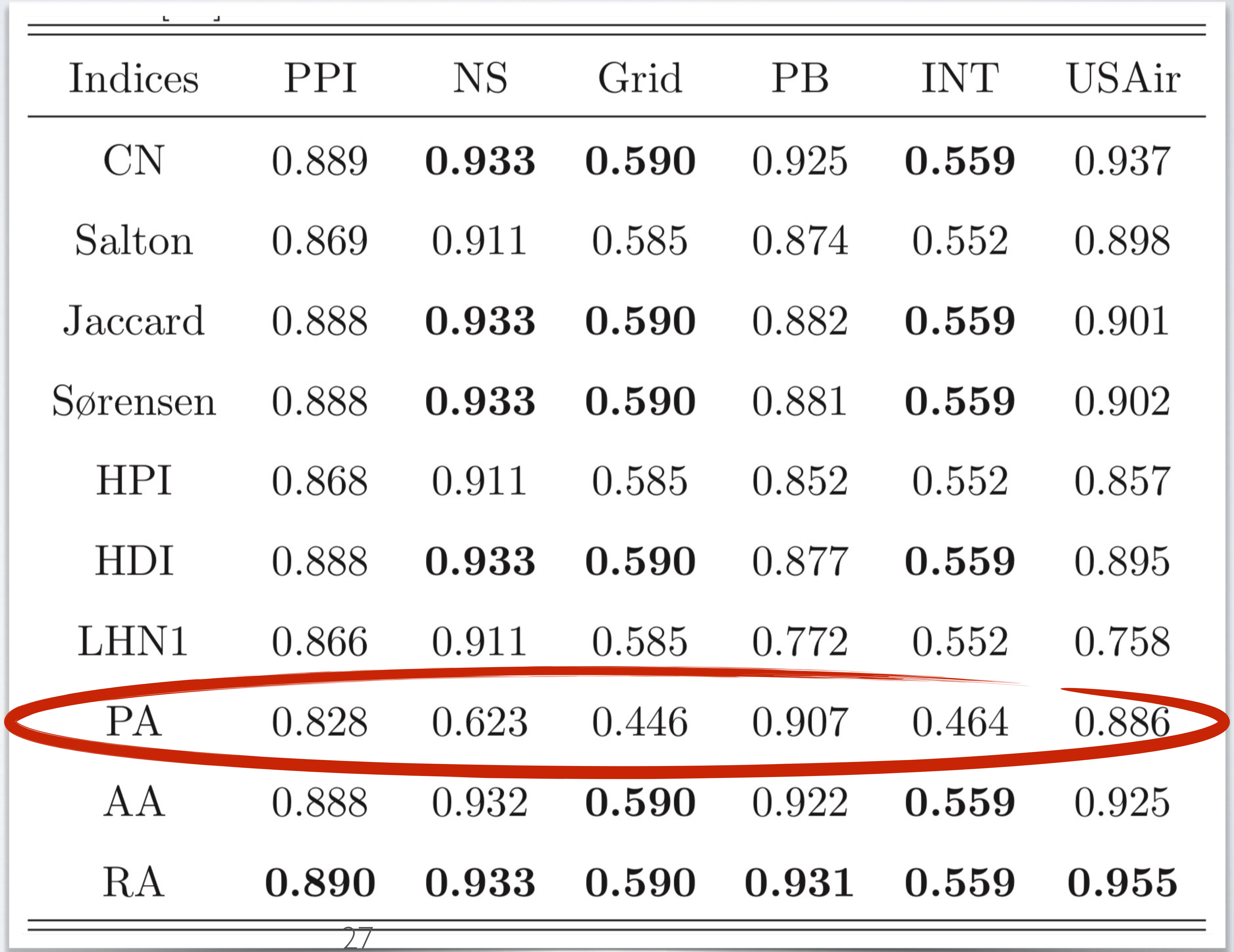


Indices	PPI	NS	Grid	PB	INT	USAir
CN	0.889	**0.933**	**0.590**	0.925	**0.559**	0.937
Salton	0.869	0.911	0.585	0.874	0.552	0.898
Jaccard	0.888	**0.933**	**0.590**	0.882	**0.559**	0.901
Sørensen	0.888	**0.933**	**0.590**	0.881	**0.559**	0.902
HPI	0.868	0.911	0.585	0.852	0.552	0.857
HDI	0.888	**0.933**	**0.590**	0.877	**0.559**	0.895
LHN1	0.866	0.911	0.585	0.772	0.552	0.758
PA	0.828	0.623	0.446	0.907	0.464	0.886
AA	0.888	0.932	**0.590**	0.922	**0.559**	0.925
RA	**0.890**	**0.933**	**0.590**	**0.931**	**0.559**	**0.955**

[Lu 2010]

WHICH ONE IS BEST?

- Compute on many networks using AUC score (Explained later)



Indices	PPI	NS	Grid	PB	INT	USAir
CN	0.889	**0.933**	**0.590**	0.925	**0.559**	0.937
Salton	0.869	0.911	0.585	0.874	0.552	0.898
Jaccard	0.888	**0.933**	**0.590**	0.882	**0.559**	0.901
Sørensen	0.888	**0.933**	**0.590**	0.881	**0.559**	0.902
HPI	0.868	0.911	0.585	0.852	0.552	0.857
HDI	0.888	**0.933**	**0.590**	0.877	**0.559**	0.895
LHN1	0.866	0.911	0.585	0.772	0.552	0.758
PA	0.828	0.623	0.446	0.907	0.464	0.886
AA	0.888	0.932	**0.590**	0.922	**0.559**	0.925
RA	**0.890**	**0.933**	**0.590**	**0.931**	**0.559**	**0.955**

[Lu 2010]

WHICH ONE IS BEST?

- All scores but PA are based on common neighbors
- =>No links between nodes at graph distance >2
- Inconsistent with observations
- =>We should combine PA and others

ML APPROACH TO LINK PREDICTION:

SIMILARITY SCORE, SUPERVISED

SUPERVISED MACHINE LEARNING

- Use Machine Learning algorithms to **learn** how to combine heuristics for optimizing predictions

- Two steps:
  - Training: show features + value to predict
  - Using/Validating: try to predict value from features

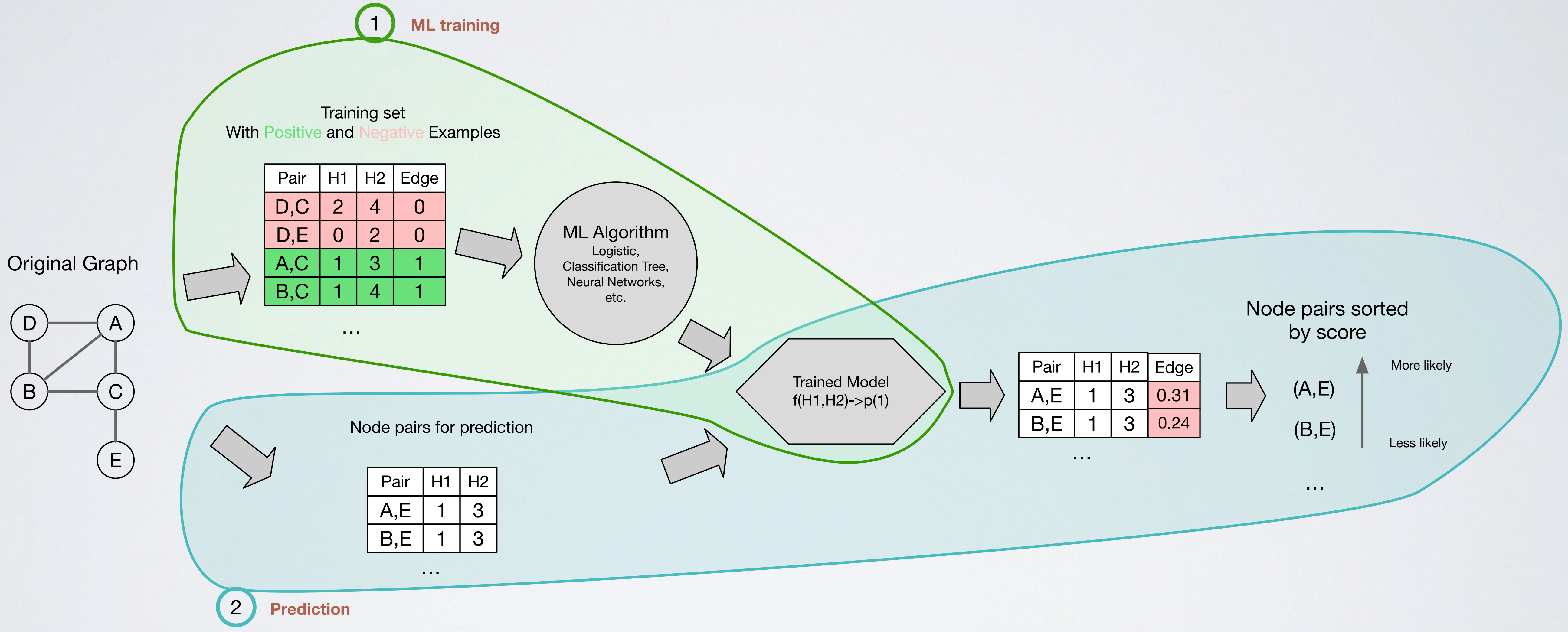
SUPERVISED MACHINE LEARNING

- Our features: similarity indices (CN, AA, PA, …)
  - **One** (limited interest) or, obviously, **several**
  - Nodes attributes can be added of available (age, salary, etc.)

- Our label/value to predict: *Link(1)* or *No link(0)* (2 **classes**)

- These types of ML algorithms are called **classifiers**
  - Logistic Classifier
  - Decision Tree Classifier
  - Neural networks Classifier
  - …

SUPERVISED MACHINE LEARNING



Training set

Pair	H1	H2	Edge
D,C	2	4	0
D,E	0	2	0
A,C	1	3	1
B,C	1	4	1

…

Node pairs for prediction

Pair	H1	H2
A,E	1	3
B,E	1	3

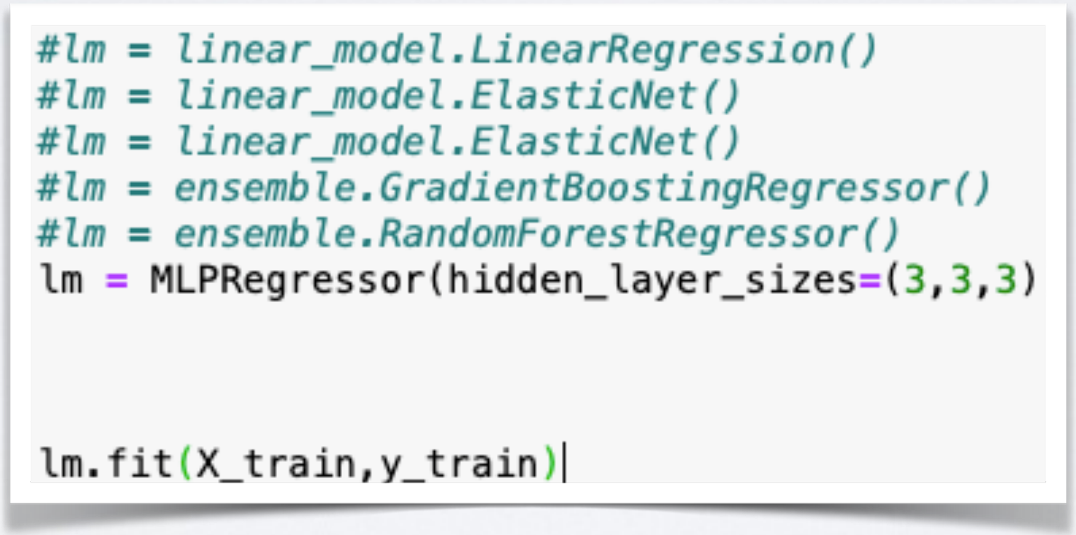
…

Pair	H1	H2	Edge
A,E	1	3	0.31
B,E	1	3	0.24

…

SUPERVISED MACHINE LEARNING

- Scores of methods, very different in their mechanisms, but same **input** and **output**



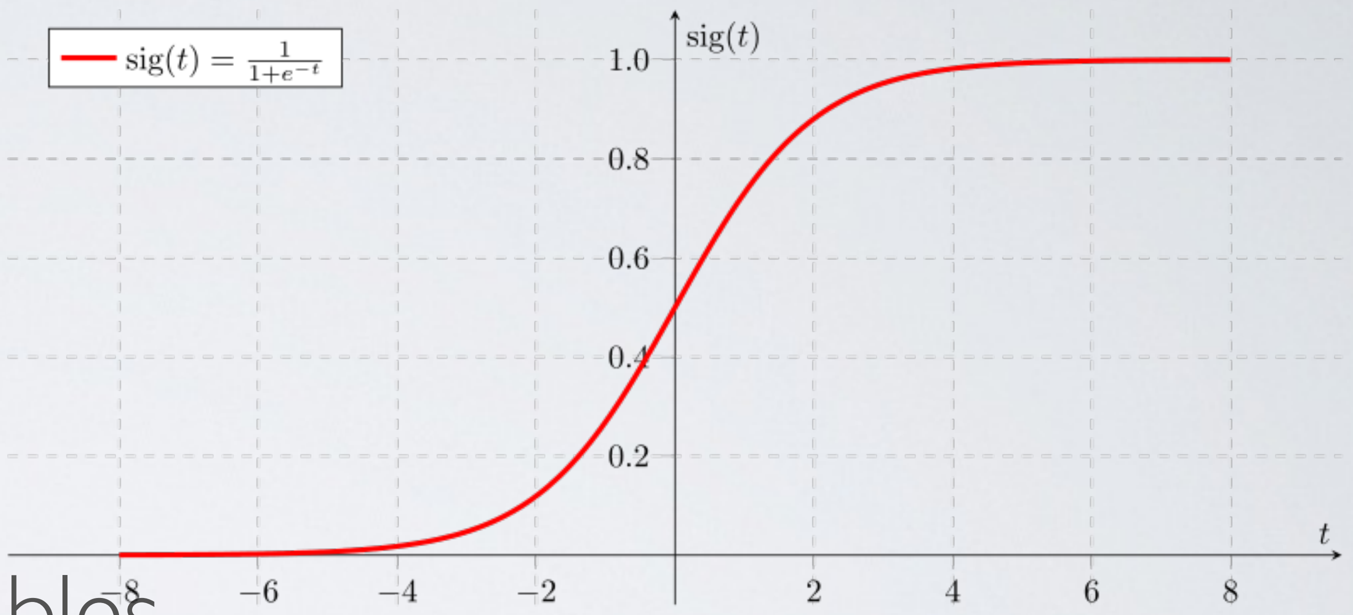
```
#lm = linear_model.LinearRegression()
#lm = linear_model.ElasticNet()
#lm = linear_model.ElasticNet()
#lm = ensemble.GradientBoostingRegressor()
#lm = ensemble.RandomForestRegressor()
lm = MLPRegressor(hidden_layer_sizes=(3,3,3))



lm.fit(X_train,y_train)
```

Let's see 2 simple examples: Logistic classification, Decision Trees https://en.wikipedia.org/wiki/Logistic_regression

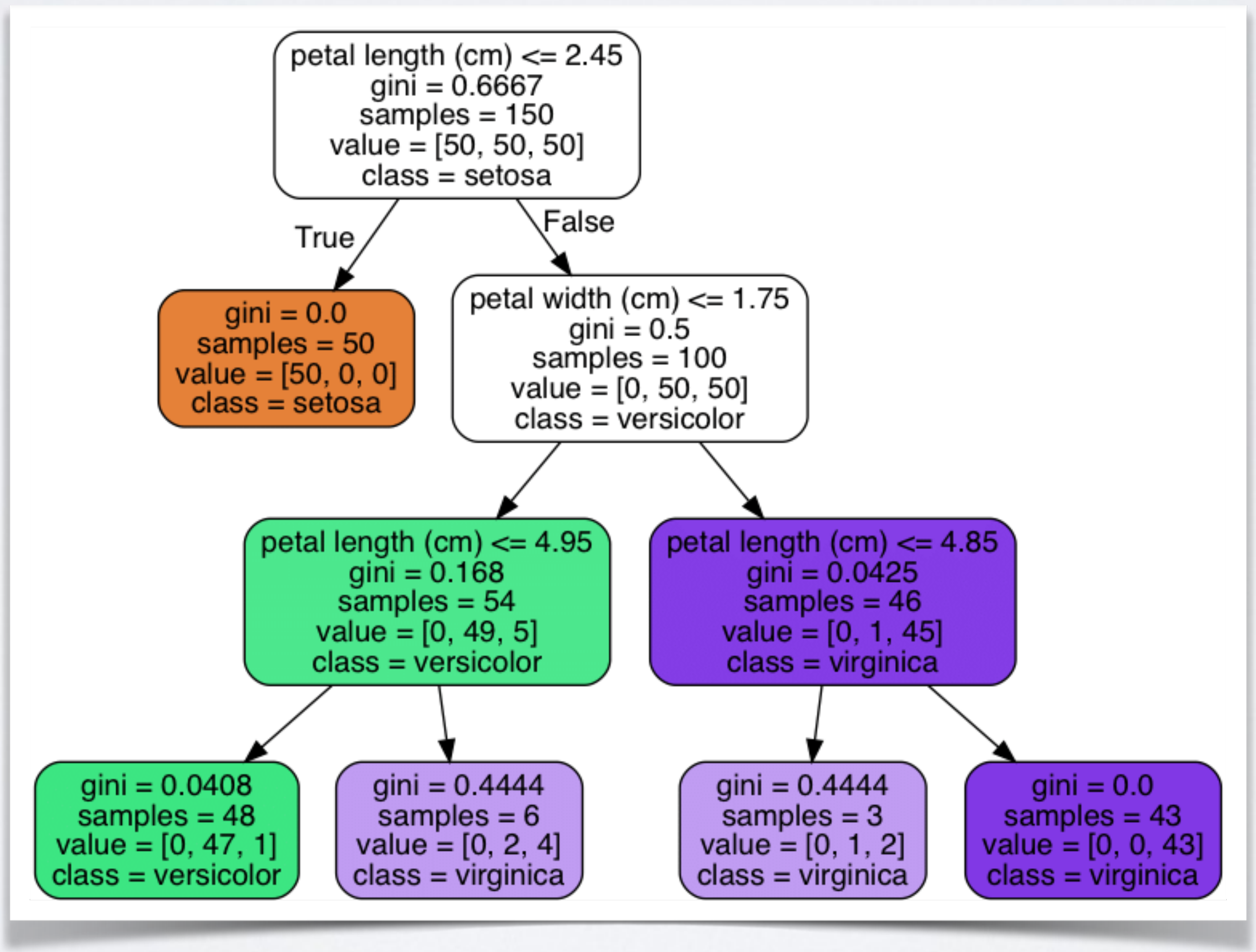
LOGISTIC CLASSIFICATION



- Value to predict y_t:
  - 0 (no edge)
  - 1 (edge)
- Linear relations between variables
  - $y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i$
- Find $\beta_0, \beta_1, \ldots$ that minimizes $y_t - y_i$

DECISION TREES

- Measure of heterogeneity (Gini, entropy…)
- Split recursively data in 2 to maximize homogeneity in child nodes



https://en.wikipedia.org/wiki/Decision_tree

DECISION TREES

- Example of possible outcomes with a decision tree:

- If CN < 1
  - IF PA>1000 => Predict 1
  - ELSE => Predict 0

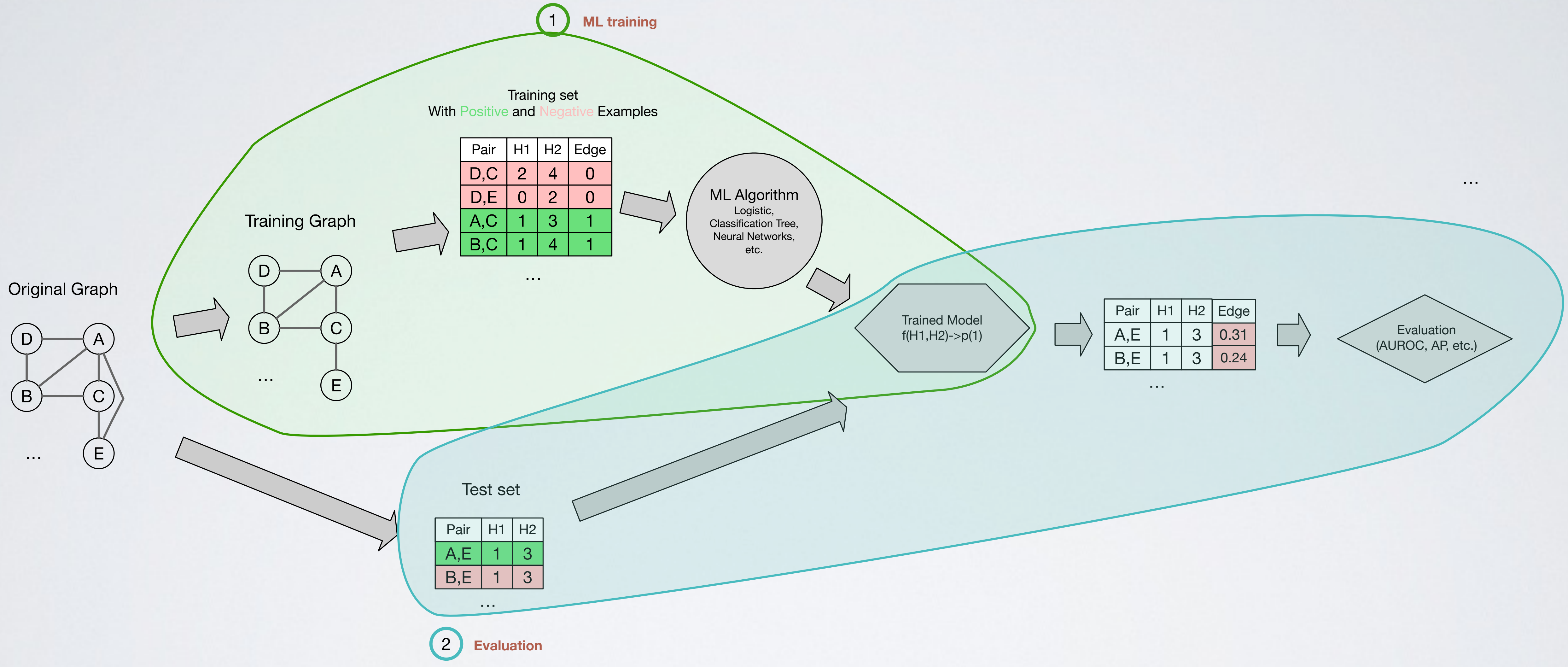
- ELSE
  - IF PA > 10000 => Predict 1
  - ELSE
    - IF AA > 10 => Predict 1
    - ELSE
    - IF JC < 0.2 => Predict 0
    - …

LINK PREDICTION EVALUATION

EVALUATION

- In order to choose a method for link prediction, it is needed to evaluate the quality of the prediction
- Several measures of prediction quality exists, but all takes the same inputs:
  - A set of test examples, and for each of them:
    - The ground truth value to predict (edge/not-edge)
    - The score provided by the prediction algorithm
  - We introduce two scores:
    - Average Precision (AP)
    - Area Under the Receiver Operating Characteristic Curve (AUROC, usually only AUC)

TRAIN/TEST SETS, SUMMARY



Pair	H1	H2	Edge
D,C	2	4	0
D,E	0	2	0
A,C	1	3	1
B,C	1	4	1

…

Pair	H1	H2
A,E	1	3
B,E	1	3

…

Pair	H1	H2	Edge
A,E	1	3	0.31
B,E	1	3	0.24

…

…

PRECISION @K

- Simple approach : Precision @k
- Fraction of correct prediction among k pairs of highest score
- Problem: which value of k to choose?
  - Affects strongly the score
  - =>Solution: a value combining scores of any k

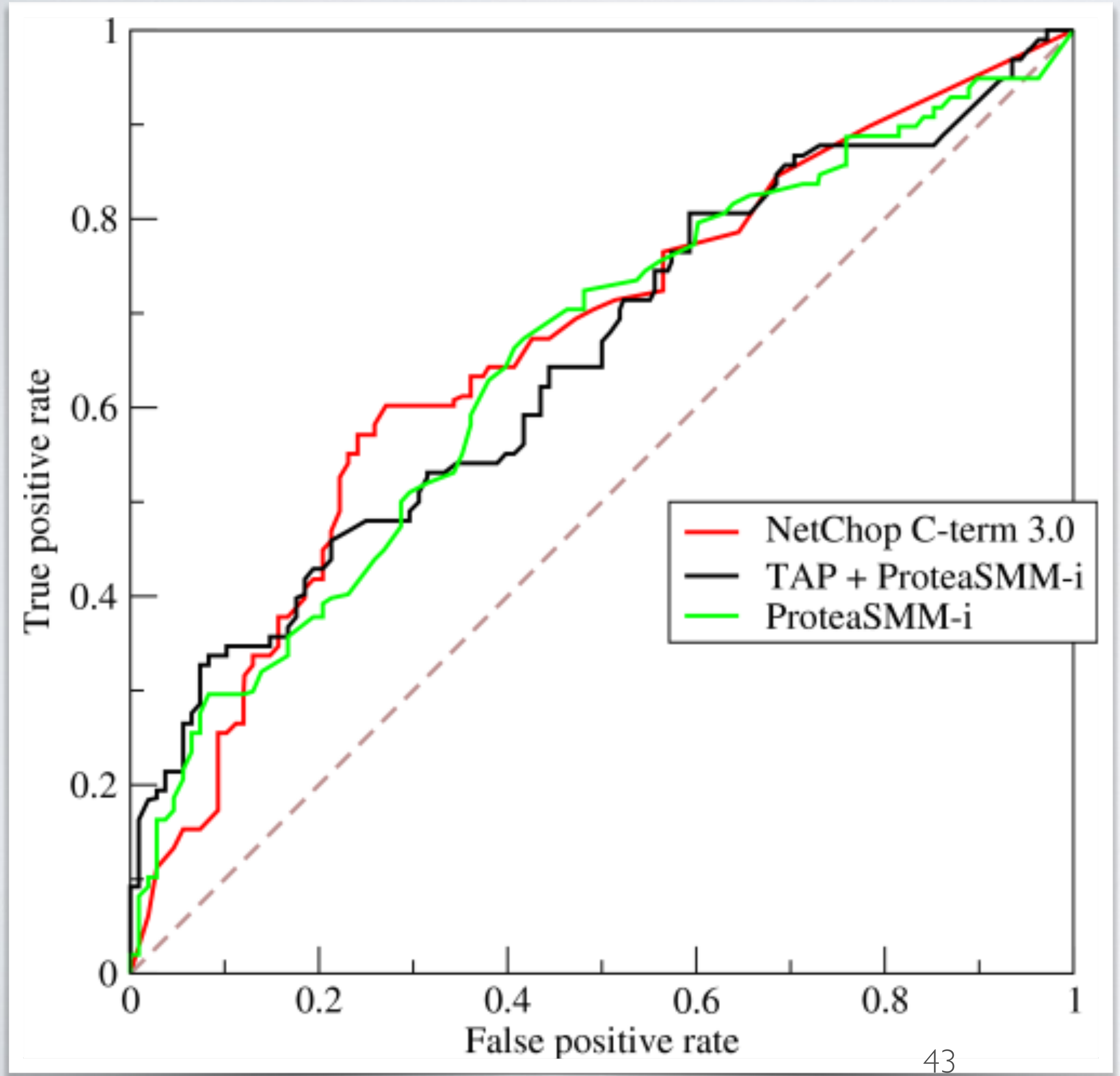
AVERAGE PRECISION

- Average Precision@k for all k
- Pros:
  - No need to arbitrarily decide k
- Property:
  - Gives higher scores to solutions making less mistakes in the beginning
    - Biased towards prediction that are good in the first few predictions

AUC - AUROC

- AUC: Area Under the Curve. Short (erroneous) name for AUROC (Area under the Receiver Operating Characteristic Curve)
- Similar idea than AP, but analyzing the relationship between
  - False positives rate (Recall)
  - True positives rate (inverse recall)
- Take the area under the curve

AUC - AUROC



True positive rate
False positive rate
0
0.2
0.4
0.6
0.8
1
NetChop C-term 3.0
TAP + ProteaSMM-i
ProteaSMM-i

AUC - AUROC

- Probabilistic interpretation:
  - If we pick a random positive example and a random negative example, probability that the positive one has a higher score
- Pros:
  - Independent on the fraction of positive examples, i.e., a balanced dataset can be used
- Cons:
  - Often very high values, (>0.95), thus small relative improvements

MICRO-VARIANT

- All these scores can also be computed node per node
  - Recommendation: most likely edge for a particular node

NODE CLASSIFICATION

Bhagat, S., Cormode, G., & Muthukrishnan, S. (2011). Node classification in social networks. In *Social network data analytics* (pp. 115-148). Springer, Boston, MA.

NODE CLASSIFICATION

- For the node classification task, we want to predict the class/ category (or numerical value) of some nodes
  - Missing values in a dataset
  - Learn to predict, in a social network/platform(Netflix…) individuals':
    - Political position, opinion on a given topic, possible security threat, …
    - Interests, tastes, etc.
    - Age, gender, sexual orientation, language spoken, salary, etc.
    - Fake accounts, spammers, bots, malicious accounts, etc.
    - …
  - Wikipedia article category, types of road in an urban network, etc.

NODE CLASSIFICATION

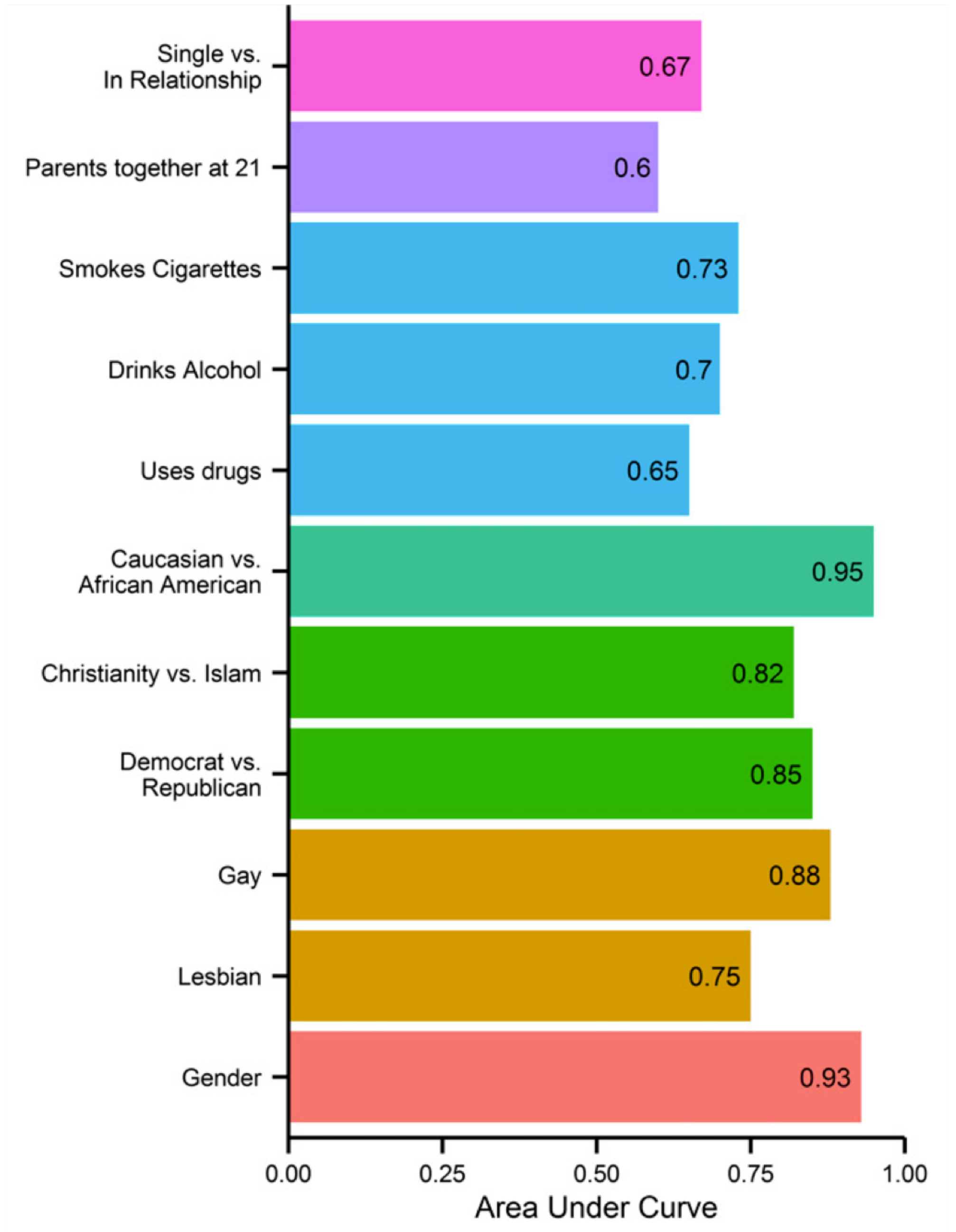


Fig. 2. Prediction accuracy of classification for dichotomous/dichotomized attributes expressed by the AUC.

Example of risks

Jernigan, C., & Mistree, B. F. (2009). Gaydar: Facebook friendships expose sexual orientation. *First Monday*, *14*(10).

Kosinski, M., Stillwell, D., & Graepel, T. (2013). Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences*, *110*(15), 5802-5805.

NODE FEATURES

- Non-network approach: Use a classification algorithm based on features of the node itself (age, salary, etc.)
- The network structure can be integrated using node centralities: Degree, clustering coefficient, betweenness, etc.
- But we can do much better:
  - "Tell me who your friends are, and I will tell you who you are"

NEIGHBORHOOD BASED CLASSIFICATION

- Classification based on the distribution of features in the neighborhood
- For each node, compute the distribution of labels in its neighborhood (vectors of length *m*, with *m* the set of all possible labels)
  - Pick the most frequent
    - e.g., political opinions
  - Train a classifier on this distribution
    - e.g., distribution of age, language in the neighborhoods to recognize bots (unexpectedly random)

NEIGHBORHOOD BASED CLASSIFICATION

- Classification based on the distribution of features in the neighborhood
- For each node, compute the distribution of labels in its neighborhood (vectors of length *m*, with *m* the set of all possible labels)
  - Pick the most frequent
    - e.g., political opinions
  - Train a classifier on this distribution
    - e.g., distribution of age, language in the neighborhoods to recognize bots (unexpectedly random)

RANDOM WALK BASED

Random Walk attributes estimation

A generalization of the previous approach consists in evaluating the distribution of attributes not only among the direct neighbors of the target nodes, but more generally among nodes that are close from it in the graph. A simple way to achieve this is to sample attributes using random walks. Several methods exist[a], for instance for a numerical attribute, the estimated value $\tilde{y_u}[c]$ for attribute c for node u can be expressed as the average value encountered by a random walk of distance t. More formally:

$$\tilde{y_u}[c] = \sum_{v \in V} p^t_{uv} v[c]$$

with p^t_{ij} the probability to encounter node v from node u after a random walk of distance t, and $v[c]$ the value of label c for node v

[a] Bhagat, Cormode, and Muthukrishnan 2011.