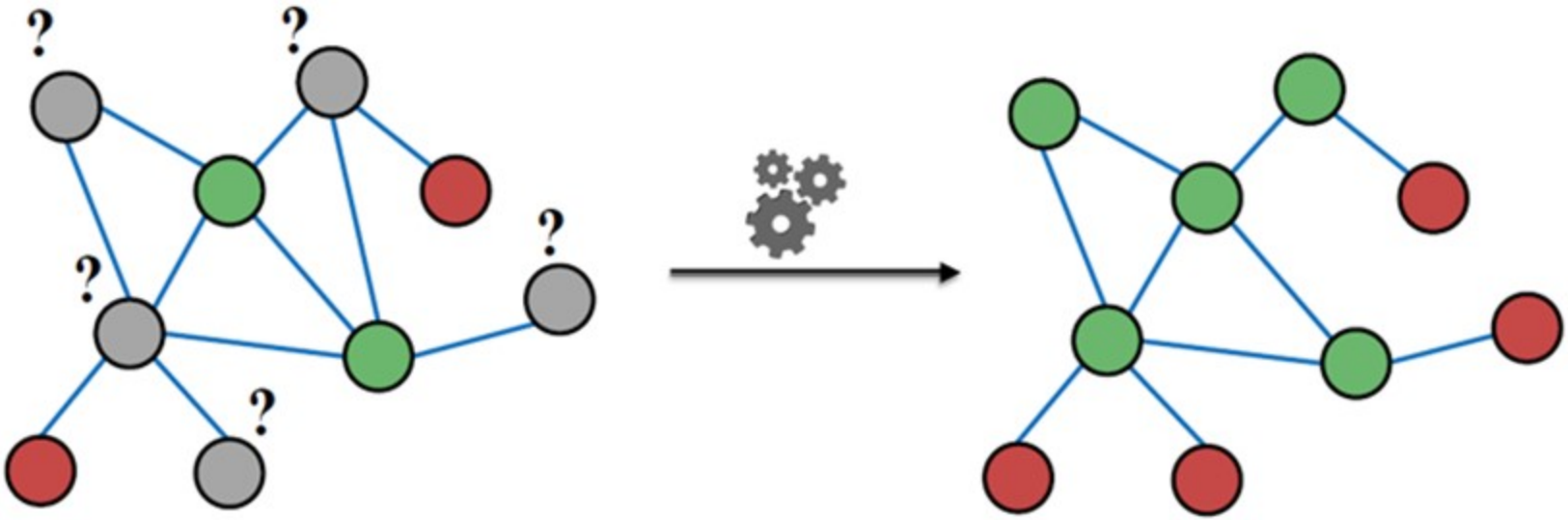
Label Propagation for Node Classification

Label Propagation for Node Classification

- Message Passing and Node Classification
- Node Correlation in networks
- Relational Classification
- Iterative Classification
- Collective Classification (Correct & Smooth)

Label Propagation for Node Classification

- Message Passing and Node Classification
  - Semi-supervised node classification



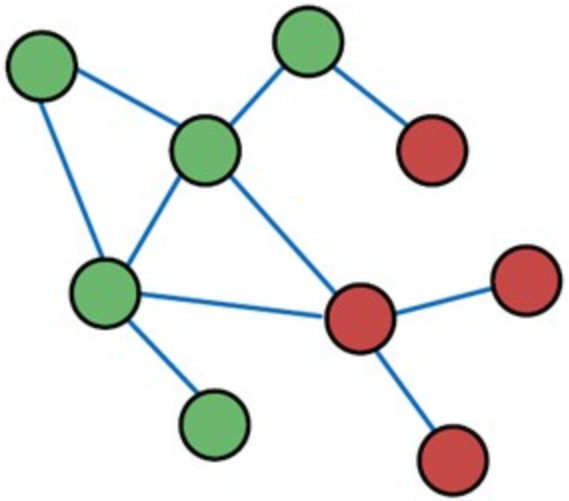
- Given labels of some nodes
- Let's predict labels of unlabeled nodes

Label Propagation for Node Classification

- Message Passing and Node Classification
  - Message passing
    - Given a network with labels on some nodes, how do we assign labels to all other nodes in the network?
    - Intuition : correlations exist in networks
    - Key concept is collective classification :
      - Assign labels to node "together"
    - Three techniques :
      - Relational Classification
      - Iterative Classification
      - Collective Classification (Correct & Smooth)

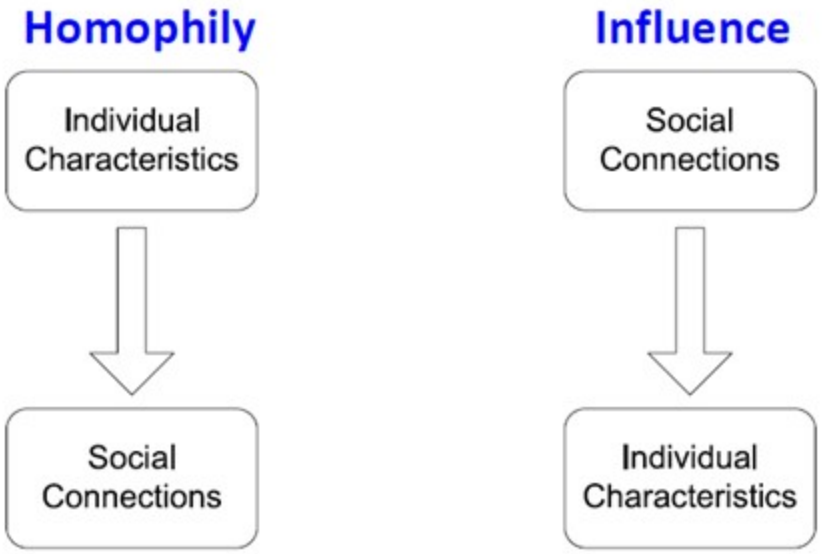
Label Propagation for Node Classification

- Node Correlation in networks
  - Behaviors of nodes are **correlated** across the links of the network
  - **Correlation**: Nearby nodes have the same color (belonging to the same class)



Label Propagation for Node Classification

- Node Correlation in networks
  - **Two explanations for why behaviors of nodes in networks are correlated:**



Label Propagation for Node Classification

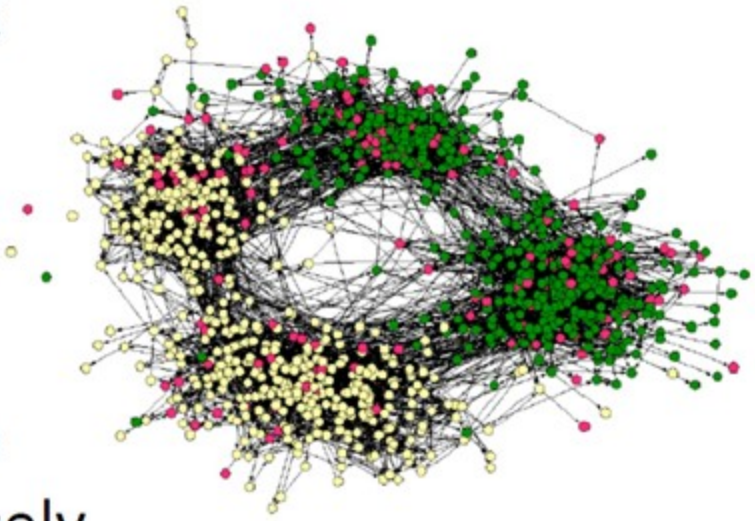
- Node Correlation in networks
  - **Homophily**: The tendency of individuals to associate and bond with similar others
    - **Example**: Researchers who focus on the same research area are more likely to establish a connection (meeting at conferences, interacting in academic talks, etc.)

Label Propagation for Node Classification

- Node Correlation in networks

Example of homophily

- Online social network
  - Nodes = people
  - Edges = friendship
  - Node color = interests (sports, arts, etc.)
- People with the same interest are more closely connected due to homophily



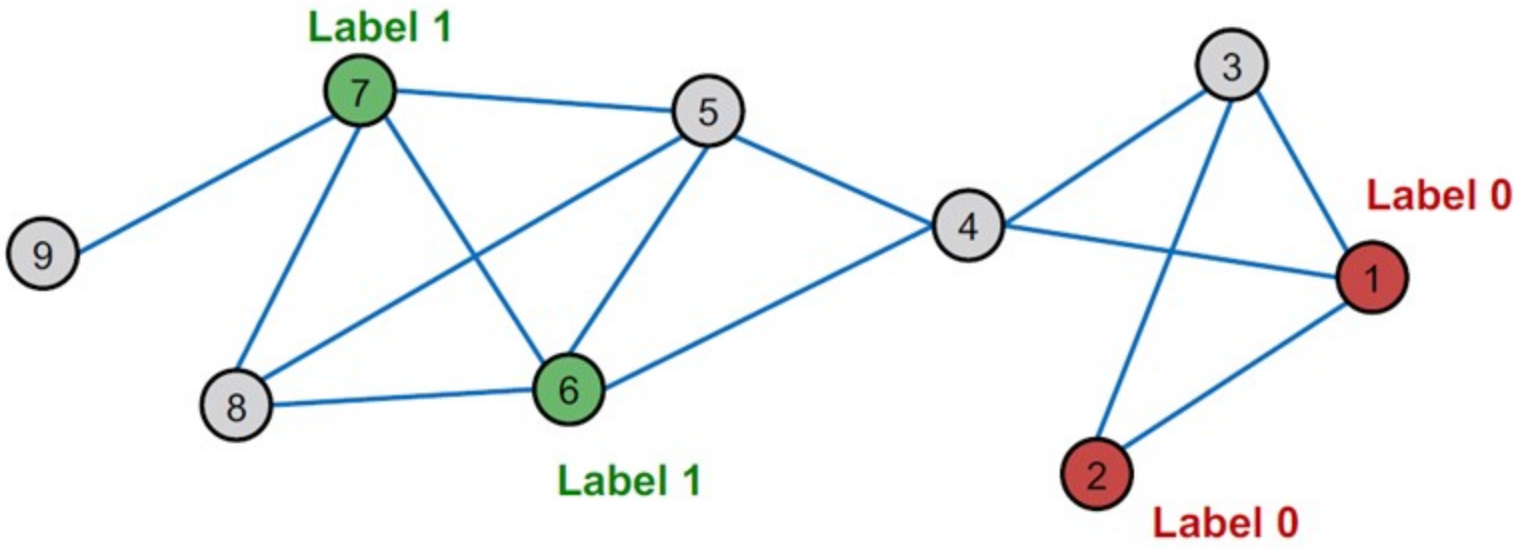
(Easley and Kleinberg, 2010)

Label Propagation for Node Classification

- Node Correlation in networks
  - **Influence**: Social connections can influence the individual characteristics of a person.
    - **Example**: I recommend my musical preferences to my friends, until one of them grows to like my same favorite genres!

Label Propagation for Node Classification

- Node Correlation in networks
  - **How do we leverage this correlation observed in networks to help predict node labels?**



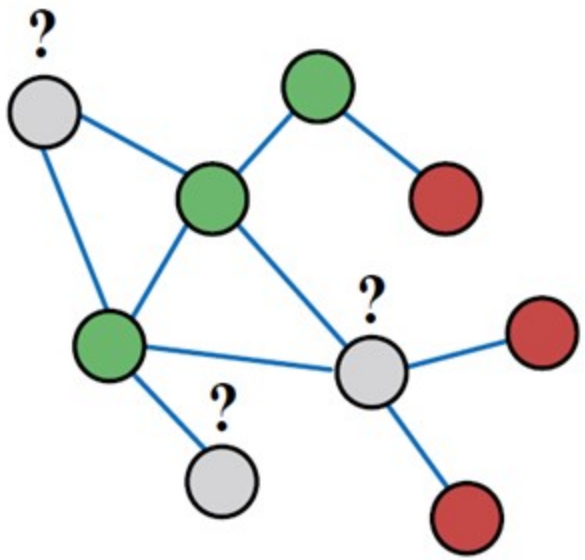
How do we predict the labels for the nodes in grey?

Label Propagation for Node Classification

- Motivations
  - **Similar nodes are typically close together or directly connected in the network:**
    - **Guilt-by-association**: If I am connected to a node with label X, then I am likely to have label X as well.
  - **Classification label** of a node v in network may depend on:
    - Features of v
    - Labels of the nodes in v's neighborhood
    - Features of the nodes in v's neighborhood

Label Propagation for Node Classification

- Semi-supervised task



Formal setting:

Given:

- Graph
- Few labeled nodes

Find: Class (red/green) of remaining nodes

Main assumption: There is homophily in the network

Label Propagation for Node Classification

- Notation
  - Let $\boldsymbol{A}$ be a $n \times n$ adjacency matrix over n nodes
  - Let $\mathrm{Y} = \{0, 1\}^n$ be a vector of **labels**:
    - $\mathrm{Y}_v = 1$ belongs to Class 1
    - $\mathrm{Y}_v = 0$ belongs to Class 0
    - There are unlabeled node needs to be classified
  - **Goal:** Predict which **unlabeled** nodes are likely **Class 1**, and which are likely **Class 0**
  - Will focus on "Semi-supervised Binary node classification"

Label Propagation for Node Classification

- Relational Classification
  - **Idea: Propagate node labels across the network**
    - Class probability Y_v of node v is a weighted average of class probabilities of its neighbors.
  - For **labeled nodes** v, initialize label Y_v with ground-truth label Y_v^*.
  - For **unlabeled nodes**, initialize $Y_v = 0.5$.
  - **Update** all nodes in a random order until convergence or until maximum number of iterations is reached.

Label Propagation for Node Classification

- Relational Classification
  - **Update** for each node v and label c (e.g. 0 or 1)

$$P(Y_v = c) = \frac{1}{\sum_{(v,u)\in E} A_{v,u}} \sum_{(v,u)\in E} A_{v,u}\, P(Y_u = c)$$

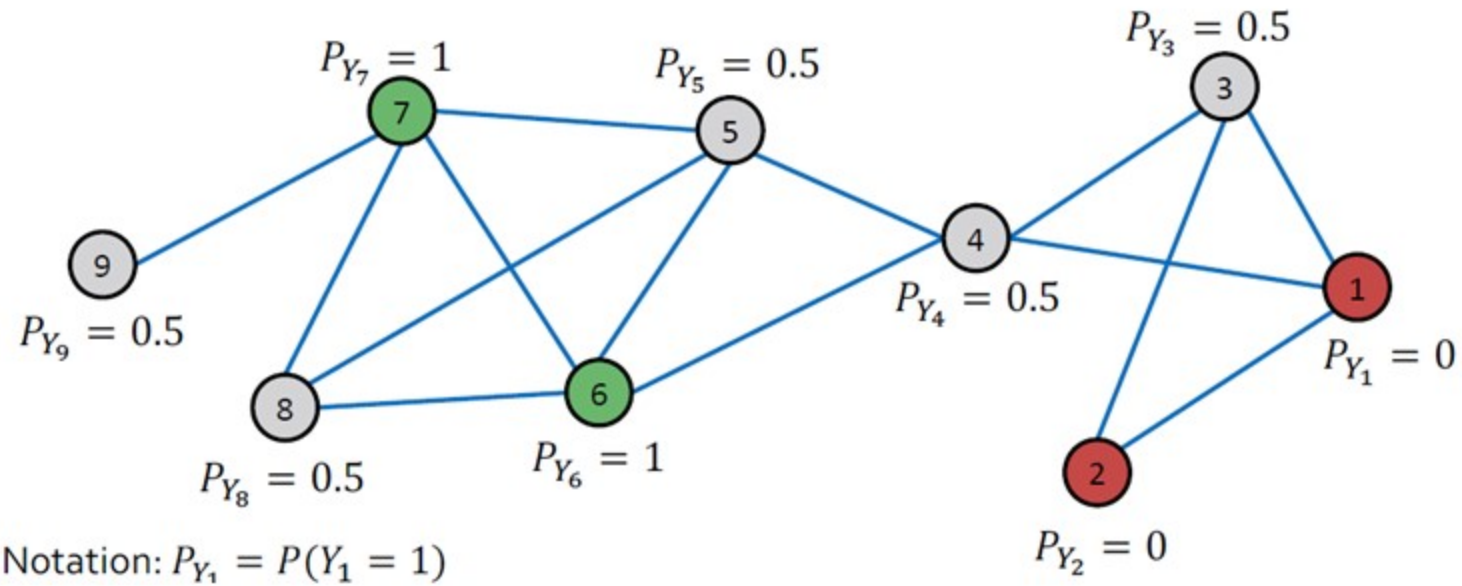
    - If edges have strength/weight information, $A_{v,u}$ can be the edge weight between v and u
    - $P(Y_v = c)$ is the probability of node v having label c
  - Challenges:
    - Convergence is not guaranteed
    - Model cannot use node feature information

Label Propagation for Node Classification

- Relational Classification : example

  Initialization:

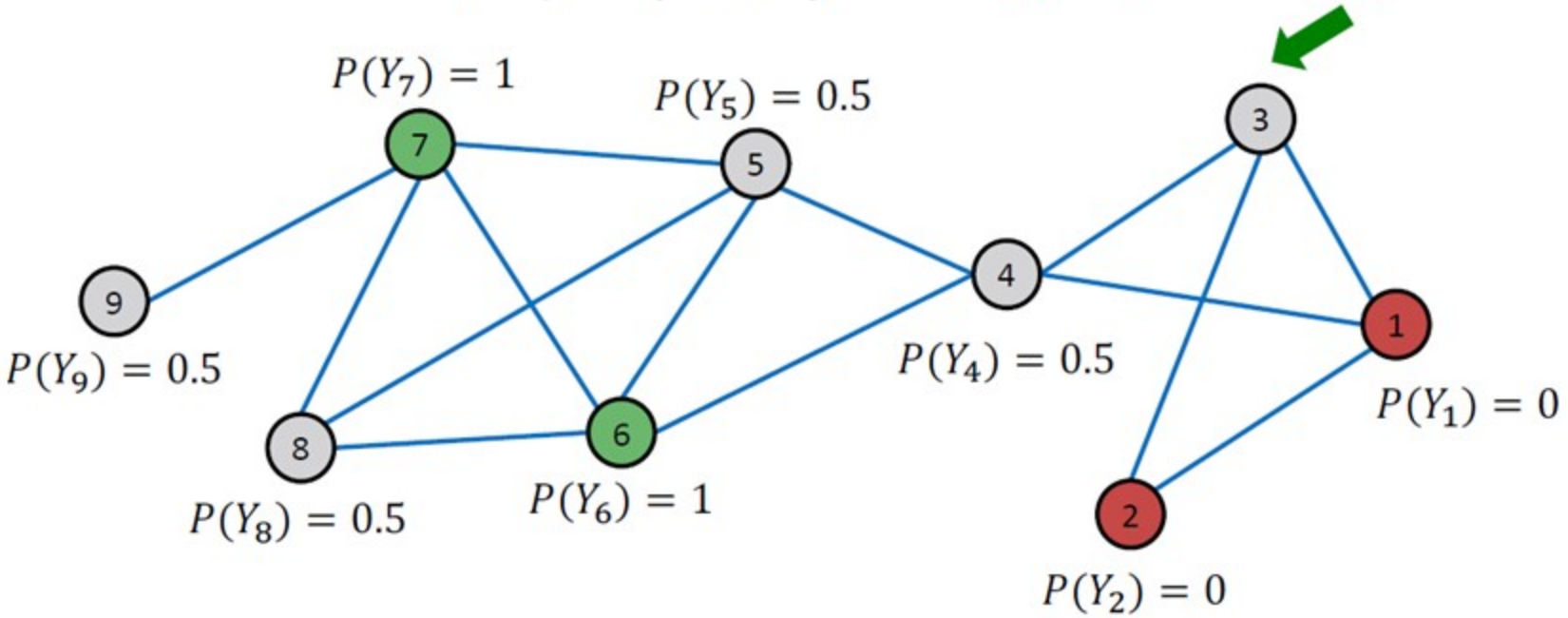
  - All labeled nodes with their labels
  - All unlabeled nodes 0.5 (belonging to class 1 with probability 0.5)



Notation: $P_{Y_1} = P(Y_1 = 1)$

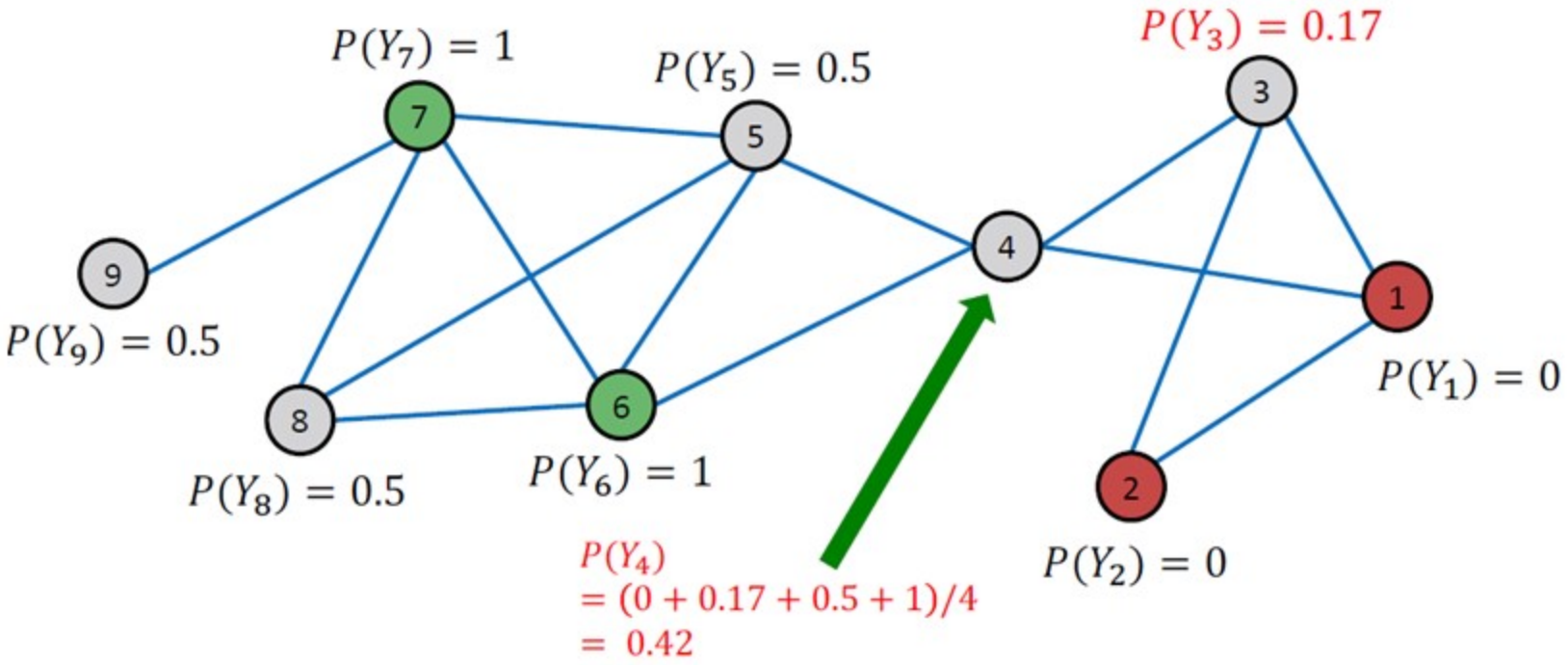
Label Propagation for Node Classification

- Relational Classification : example
  - Update for the 1st Iteration:
    - For node 3, $N_3 = \{1, 2, 4\}$



Label Propagation for Node Classification

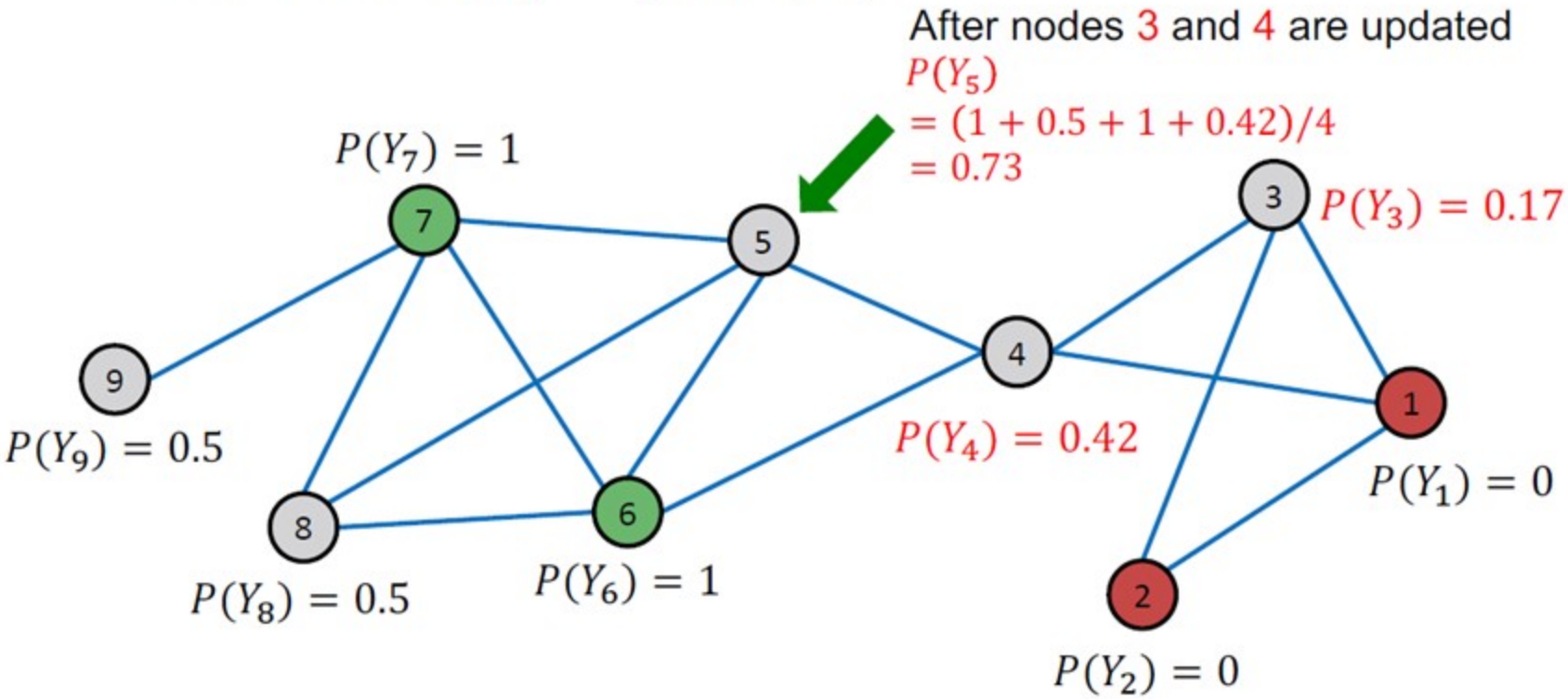
- Relational Classification : example
  - Update for the 1st Iteration:
    - For node 4, $N_4 = \{1, 3, 5, 6\}$



After Node 3 is updated

Label Propagation for Node Classification

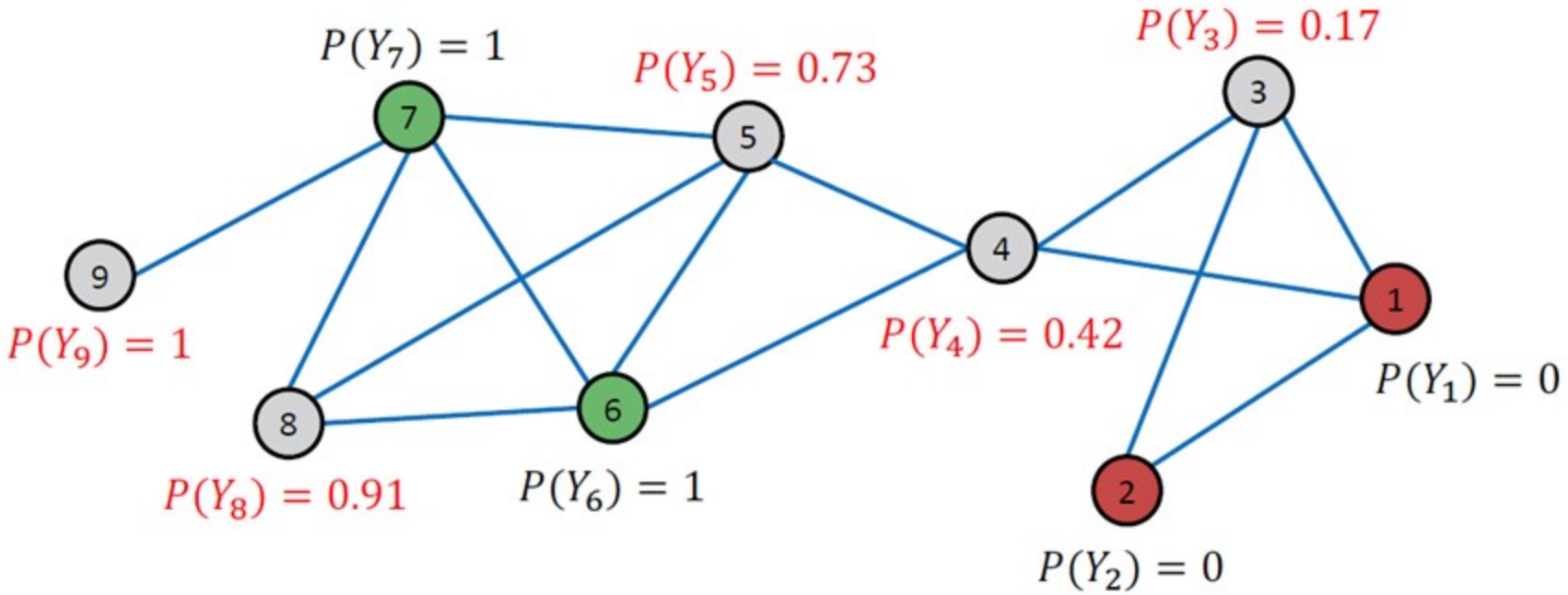
- Relational Classification : example
  - Update for the 1st Iteration:
    - For node 5, $N_5 = \{4, 6, 7, 8\}$



Label Propagation for Node Classification

- Relational Classification : example

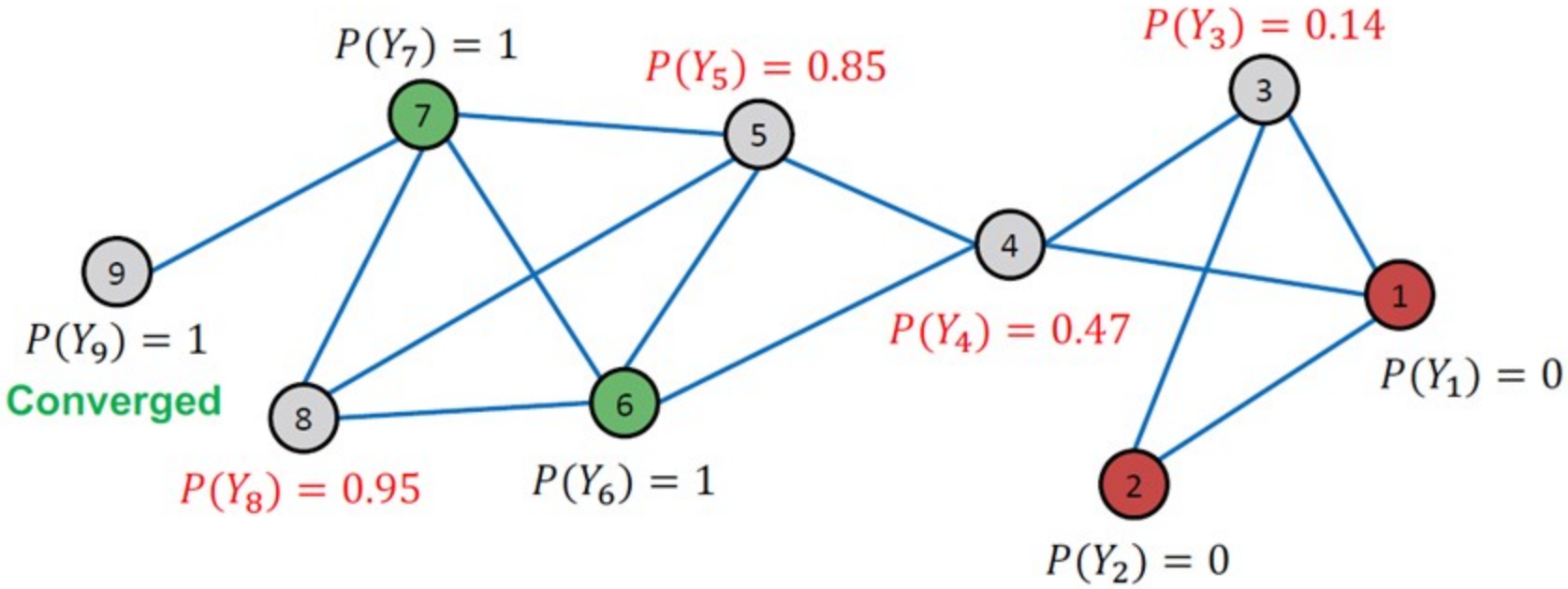
  After Iteration 1 (a round of updates for all unlabeled nodes)



Label Propagation for Node Classification

- Relational Classification : example

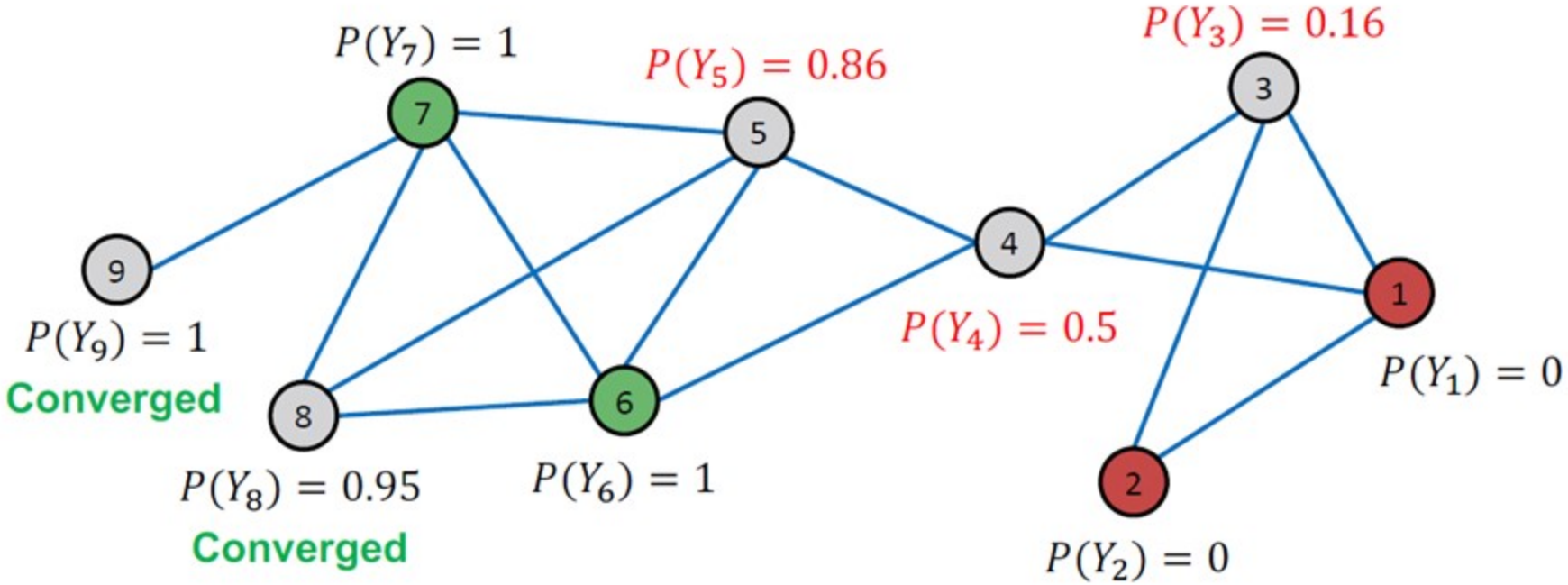
After Iteration 2



Label Propagation for Node Classification

- Relational Classification : example

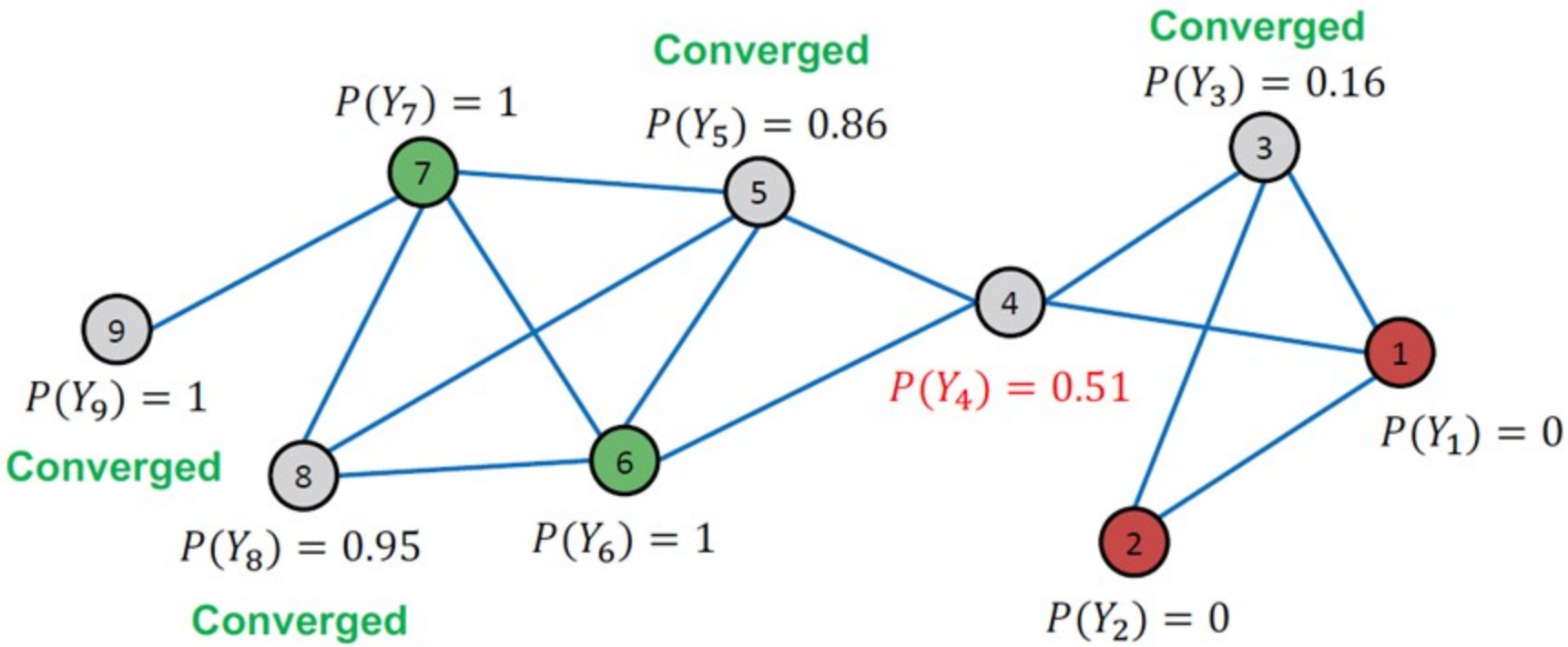
After Iteration 3



Label Propagation for Node Classification

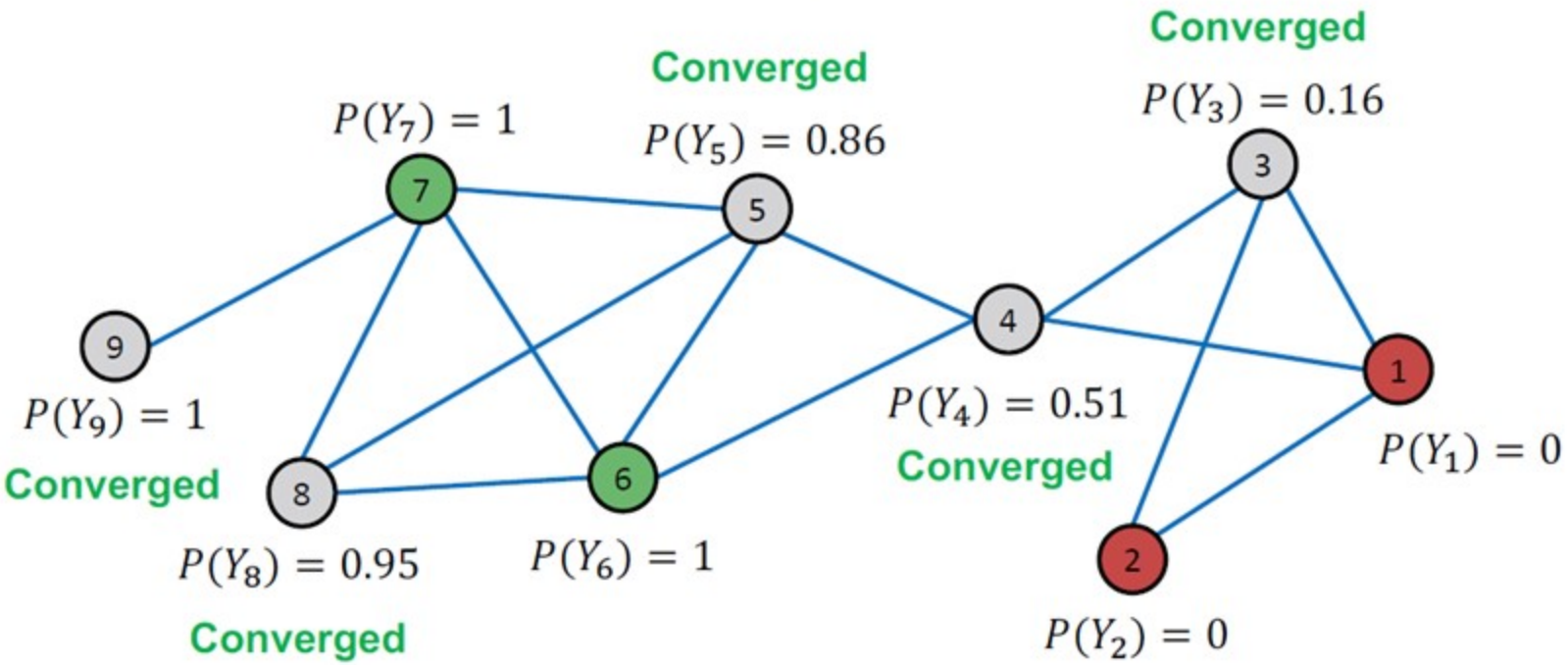
- Relational Classification : example

  After Iteration 4



Label Propagation for Node Classification

- Relational Classification : example
  - All scores stabilize after 4 iterations. We therefore predict:
    - **Nodes 4, 5, 8, 9 belong to class 1** ($P_{Y_v} > 0.5$)
    - **Nodes 3 belong to class 0** ($P_{Y_v} < 0.5$)



Label Propagation for Node Classification

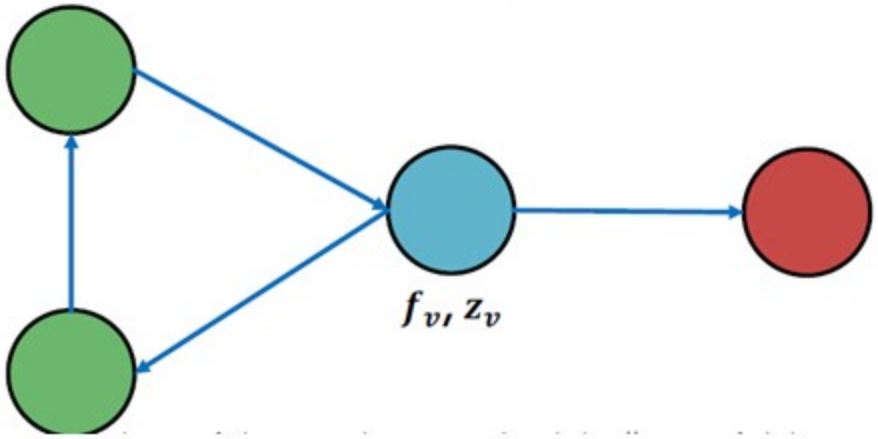
- Iterative Classification
  - **Main idea of iterative classification:** Classify node v based on its **attributes f_v** as well as **labels z_v** of neighbor set N_v.
  - **Input: Graph**
    - f_v : feature vector for node v
    - Some nodes v are labeled with Y_v
  - **Task:** Predict label of unlabeled nodes
  - **Approach: Train two classifiers:**
    - $\phi_1(f_v)$ = Predict node label based on node feature vector f_v. This is called **base classifier.**
    - $\phi_2(f_v, z_v)$ = Predict label based on node feature vector f_v and summary z_v of labels of v's neighbors. This is called **relational classifier**.

Label Propagation for Node Classification

- Iterative Classification

  How do we compute the summary z_v of labels of v's neighbors N_v?

  - z_v = **vector that captures labels around node v**
    - Histogram of the number (or fraction) of each label in N_v
    - Most common label in N_v
    - Number of different labels in N_v



Label Propagation for Node Classification

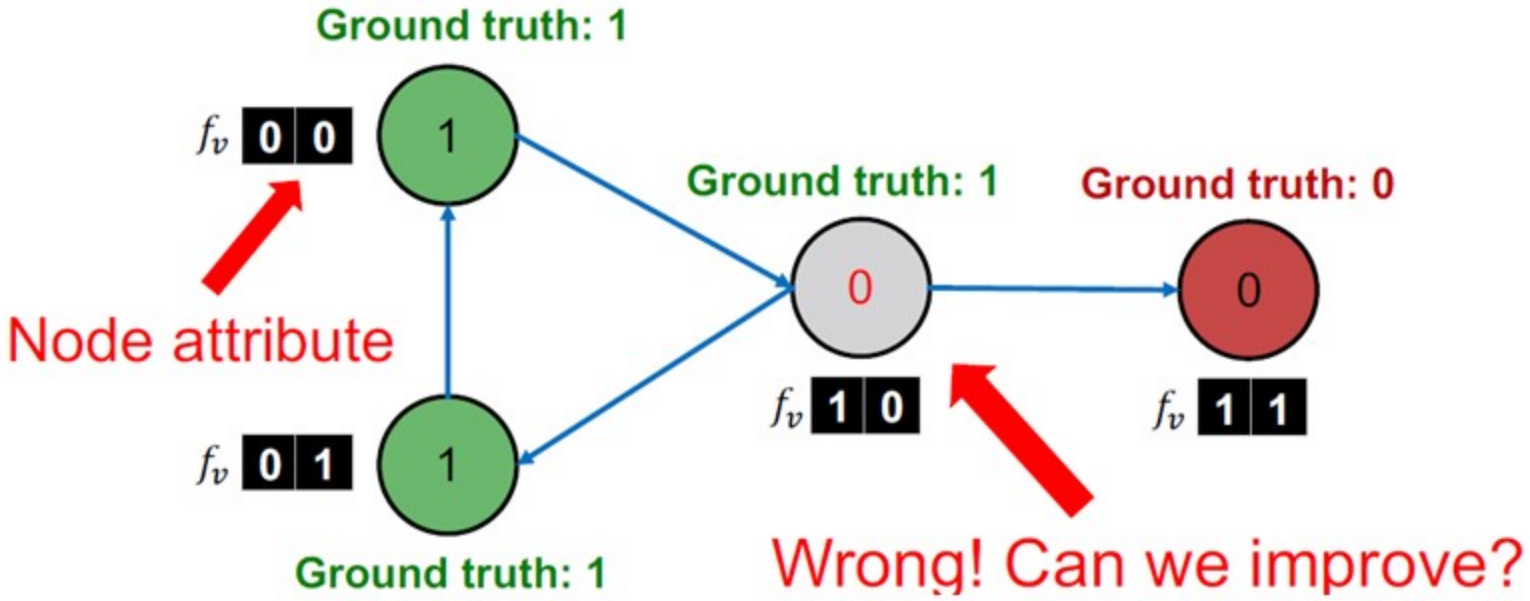
- Iterative Classification
  - **Phase 1: Classify based on node attributes alone**
    - On the labeled **training set**, train two classifiers:
      - **Base classifier:** $\phi_1(f_v)$ to predict Y_v based on f_v
      - **Relational classifier:** $\phi_2(f_v, z_v)$ to predict Y_v based on f_v and summary z_v of labels of v's neighbors
  - **Phase 2: Iterate till convergence**
    - On **test set**, set labels Y_v based on the classifier ϕ_1, compute z_v and predict the labels with ϕ_2
    - **Repeat** for each node v:
      - Update z_v based on Y_u for all $u \in N_v$
      - Update Y_v based on the new z_v (ϕ_2)
    - Iterate until class labels stabilize or max number of iterations is reached
    - **Note:** Convergence is not guaranteed

Label Propagation for Node Classification

- Iterative Classification : example (web page classif)
  - **Input:** Graph of web pages
  - **Node:** Web page
  - **Edge:** Hyper-link between web pages
    - **Directed edge:** a page points to another page
  - **Node features:** Webpage description
    - For simplicity, we only consider two binary features
  - **Task:** Predict the topic of the webpage

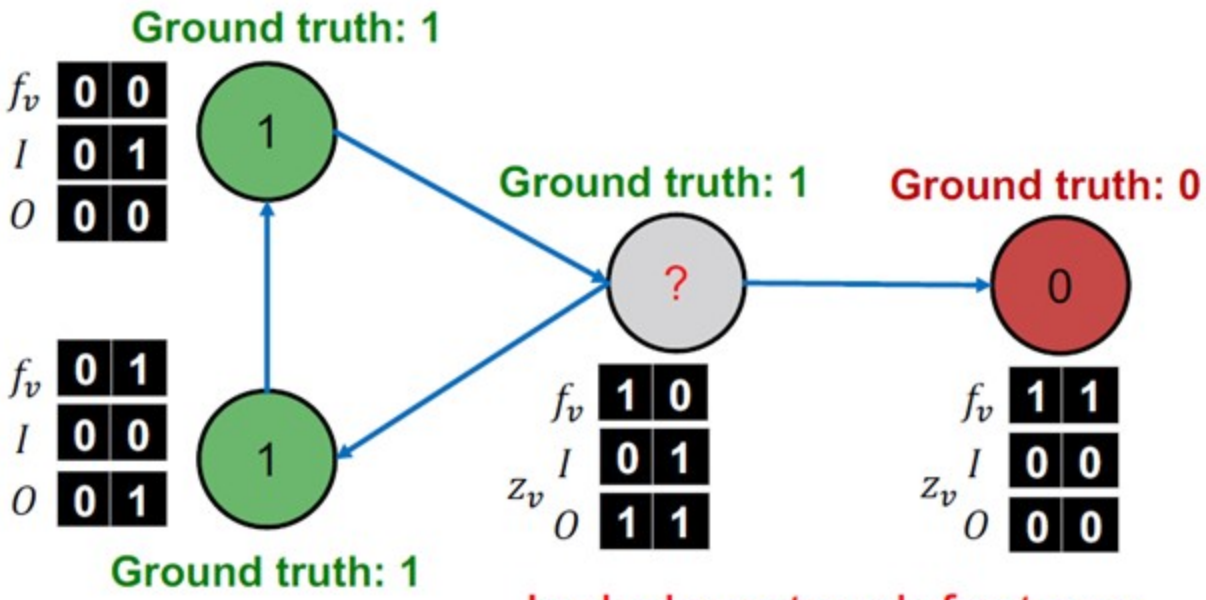
Label Propagation for Node Classification

- Iterative Classification : example (web page classif)
  - Baseline: Train a classifier (e.g., linear classifier) to classify pages based on node attributes.



Label Propagation for Node Classification

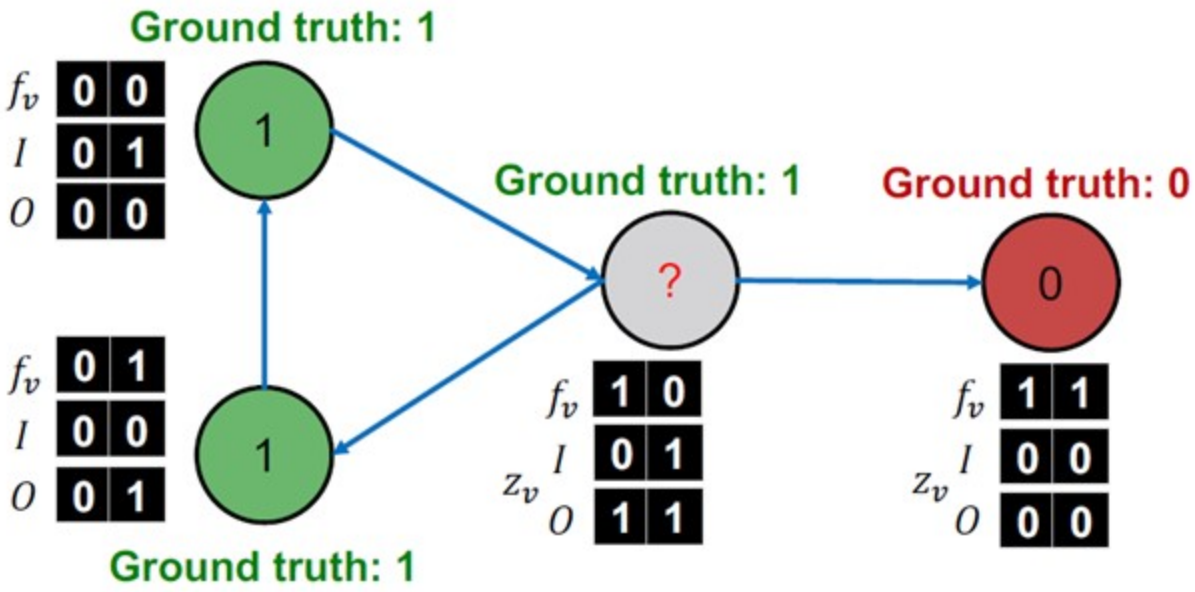
- Iterative Classification : example (web page classif)
  - Each node maintains vectors z_v of neighborhood labels:
    - I = **Incoming** neighbor label information vector.
    - O = **Outgoing** neighbor label information vector.
      - $I_0 = 1$ if at least one of the incoming pages is labelled 0.
        Similar definitions for I_1, O_0, and O_1



Label Propagation for Node Classification

- Iterative Classification : example (web page classif)
  - On **training labels**, train two classifiers:
    - Node attribute vector only: $\phi_1(f_v)$
    - Node attribute and link vectors z_v: $\phi_2(f_v, z_v)$

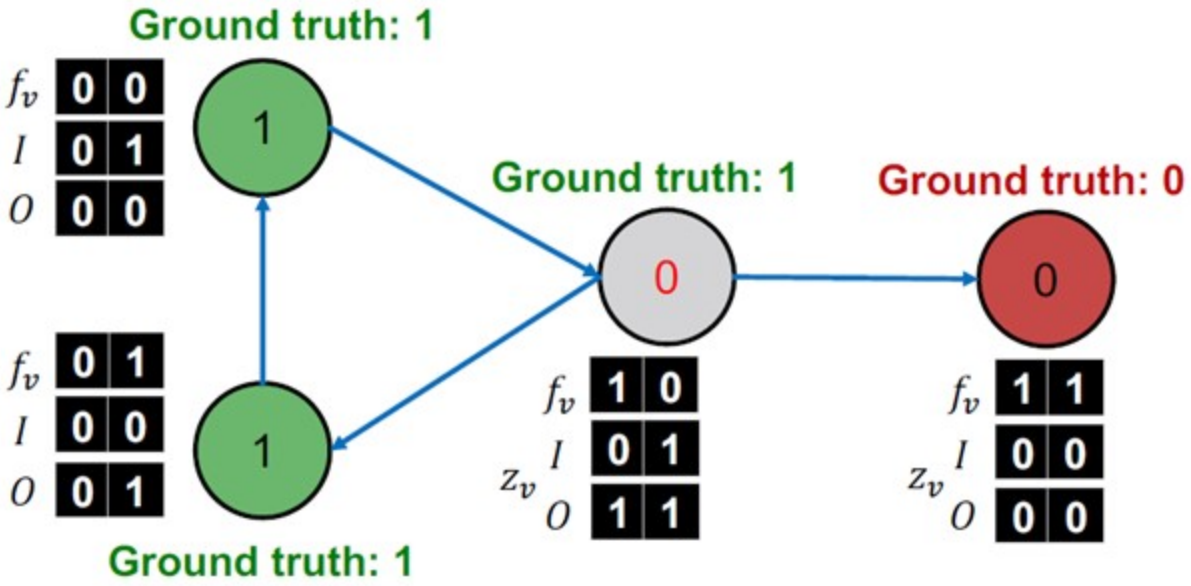
1. Train classifiers
2. Apply classifier to unlab. set
3. Iterate
   4. Update relational features z_v
   5. Update label Y_v



Label Propagation for Node Classification

- Iterative Classification : example (web page classif)
  - On the **unlabeled set**:
    - Use trained node feature vector classifier ϕ_1 to set Y_v

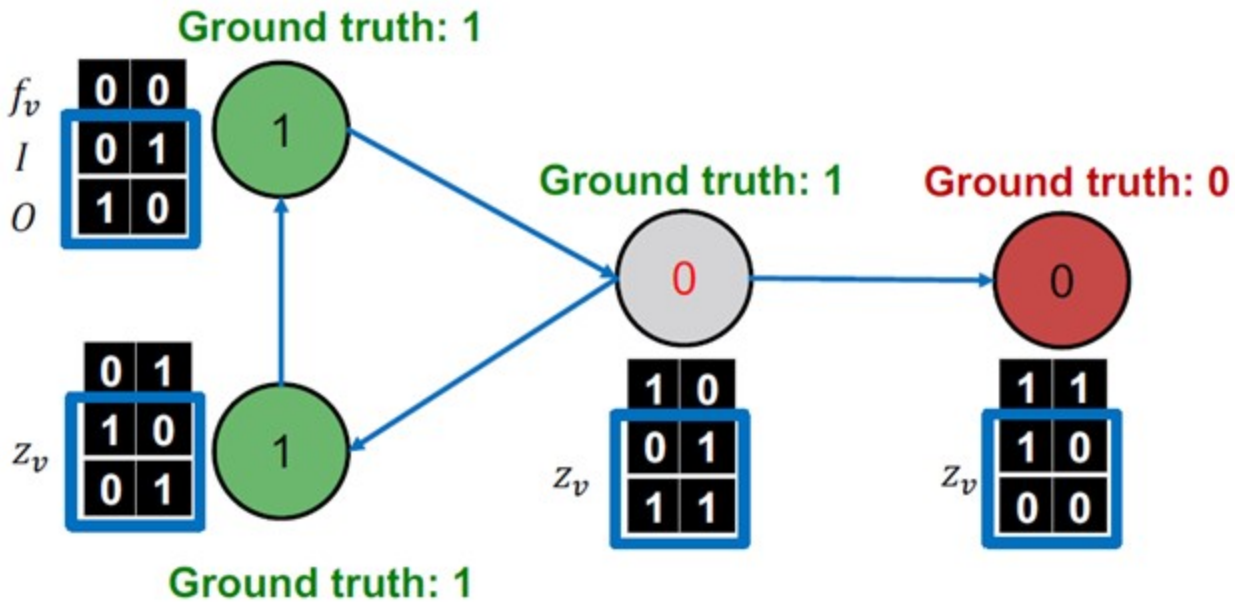
1. Train classifiers
2. **Apply classifier to unlab. set**
3. Iterate
   4. Update relational features z_v
   5. Update label Y_v



Label Propagation for Node Classification

- Iterative Classification : example (web page classif)
  - **Update z_v for all nodes:**

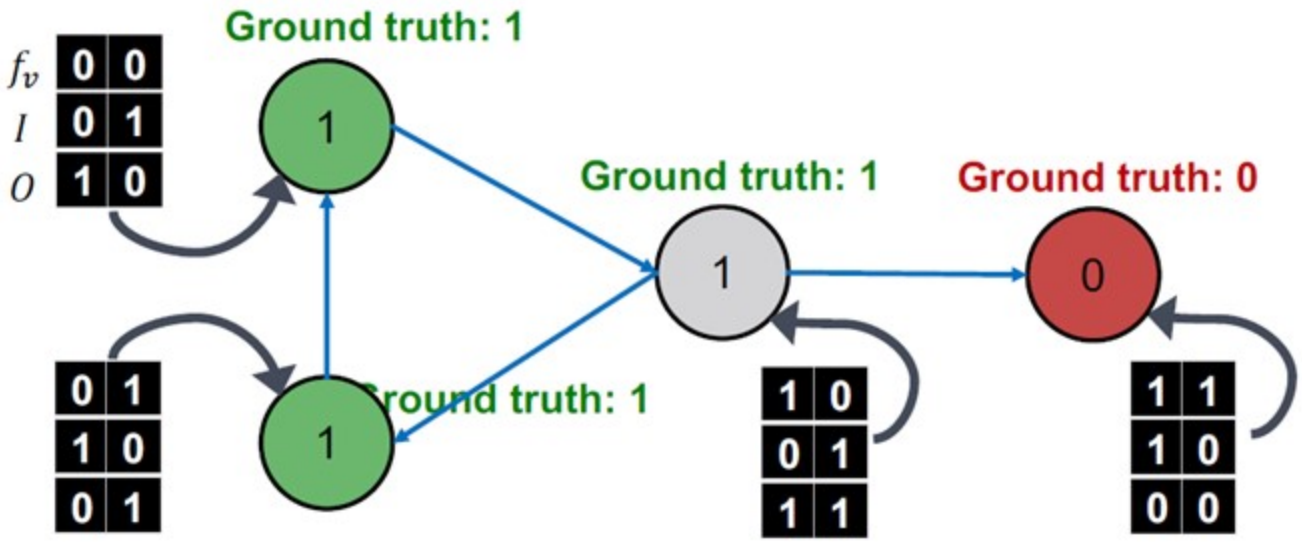
1. Train classifiers
2. Apply classifier to unlab. set
3. Iterate
   4. Update relational features z_v
   5. Update label Y_v



Label Propagation for Node Classification

- Iterative Classification : example (web page classif)
  - **Re-classify all nodes with ϕ_2:**

1. Train classifiers
2. Apply classifier to unlab. set
3. Iterate
   4. Update relational features z_v
   5. Update label Y_v

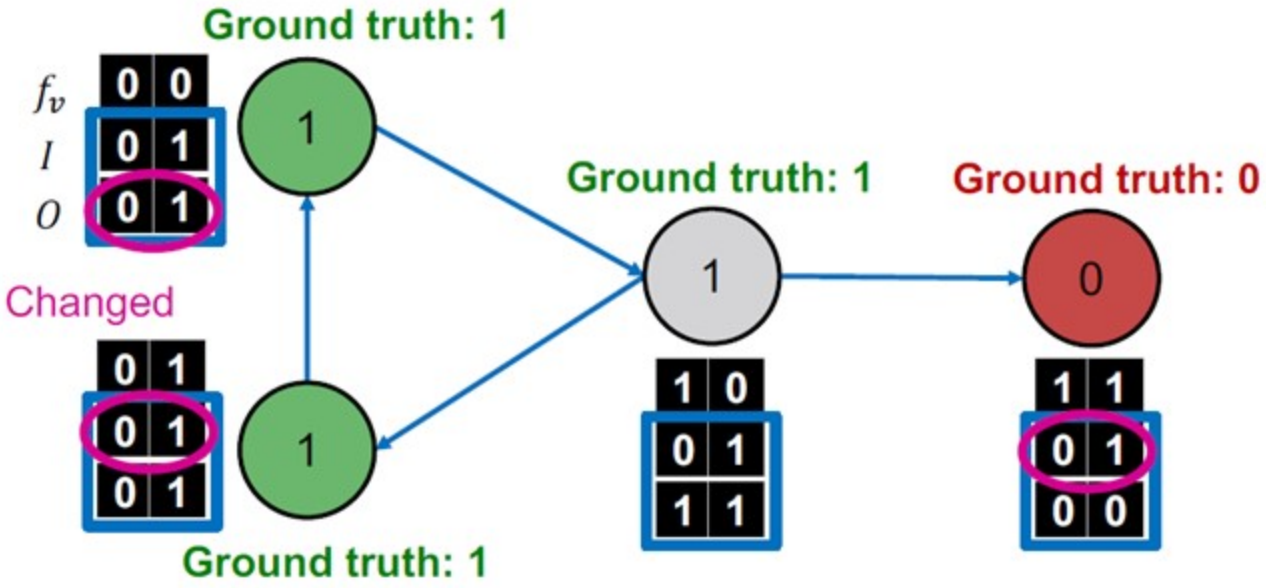


Now it's correct prediction!

Label Propagation for Node Classification

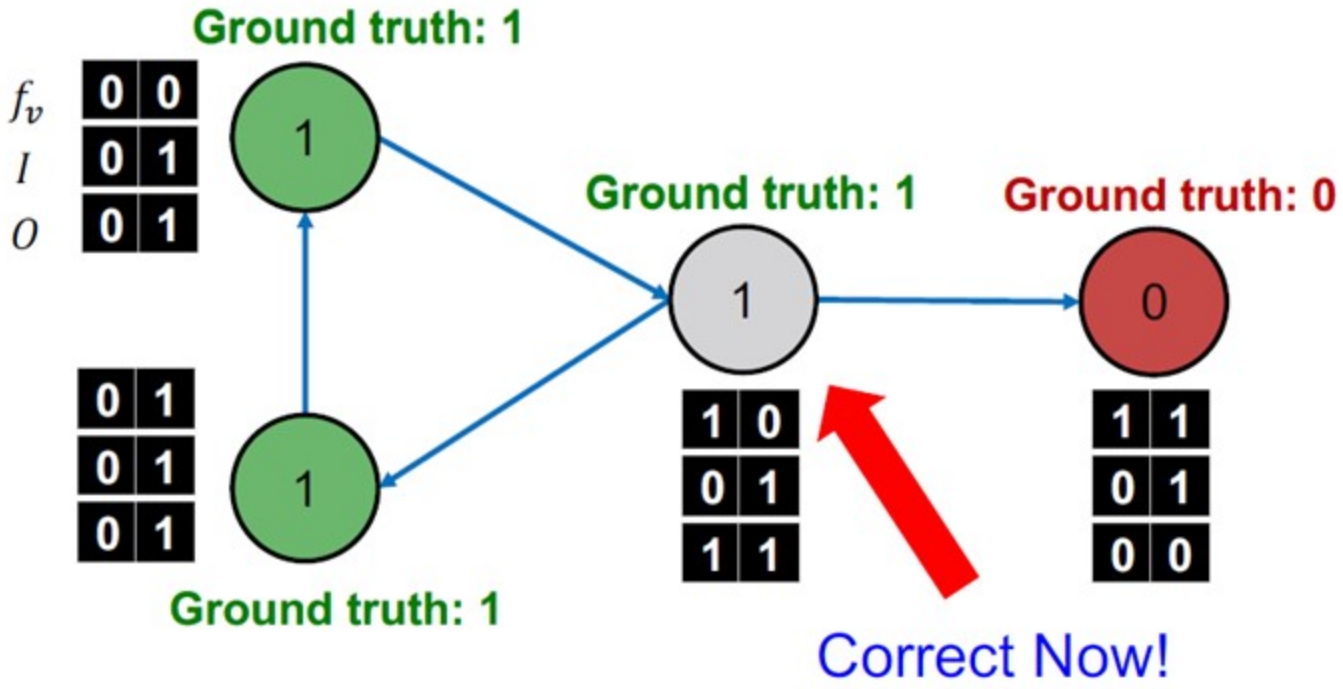
- Iterative Classification : example (web page classif)
  - **Continue until convergence**
    - Update z_v based on Y_v
    - Update $Y_v = \phi_2(f_v, z_v)$

1. Train classifiers
2. Apply classifier to unlab. set
3. Iterate
   4. Update relational features z_v
   5. Update label Y_v



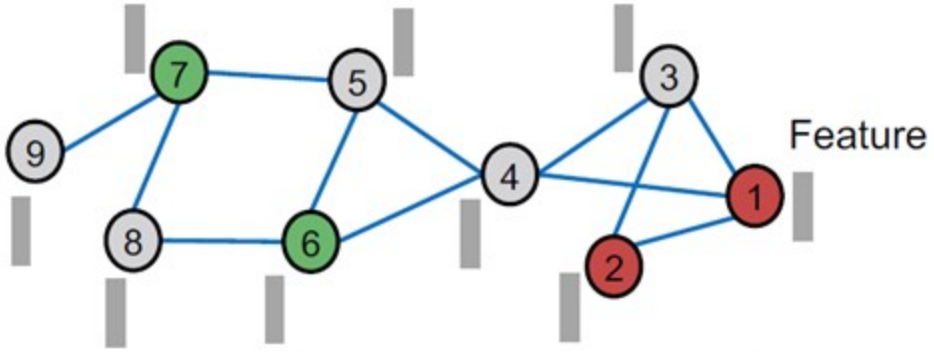
Label Propagation for Node Classification

- Iterative Classification : example (web page classif)
  - **Stop iteration**
    - After convergence or when maximum iterations are reached



Label Propagation for Node Classification

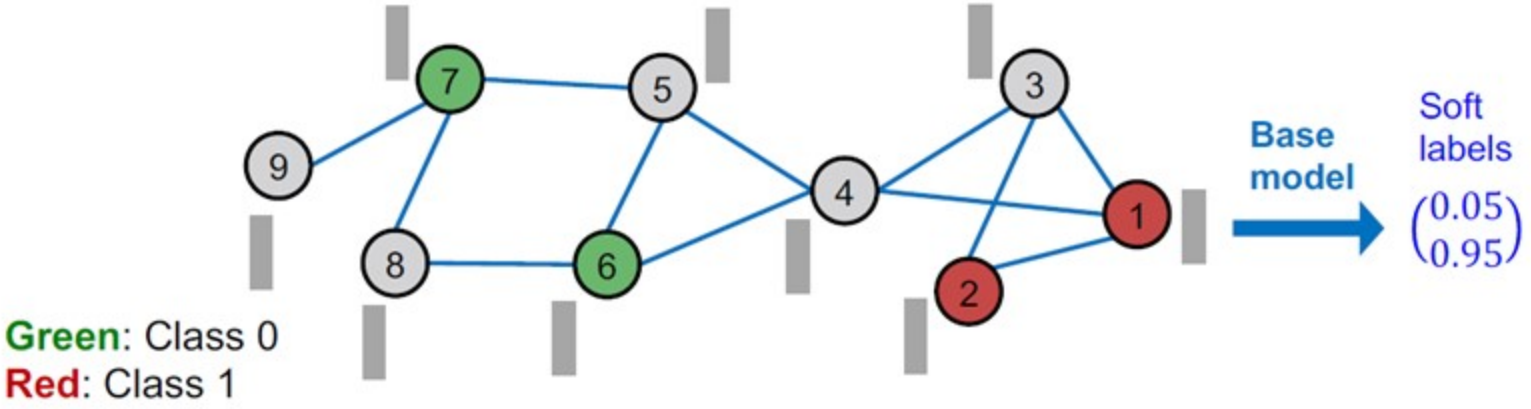
- Collective Classification (Correct & Smooth)
  - **Setting:** A partially labeled graph and features over nodes.



  - **C&S follows the three-step procedure:**
    1. Train base predictor
    2. Use the base predictor to predict soft labels of all nodes.
    3. **Post-process the predictions using graph structure** to obtain the final predictions of all nodes.

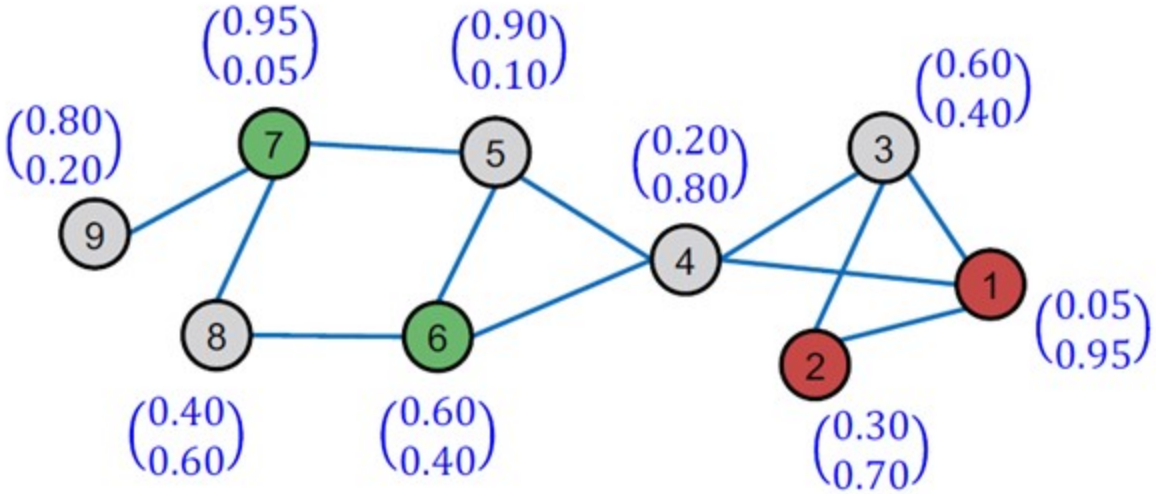
Label Propagation for Node Classification

- Collective Classification (Correct & Smooth)
  - **(1) Train a base predictor that predict soft labels (class probabilities) over all nodes.**
    - Labeled nodes are used for train/validation data.
    - Base predictor can be simple:
      - Linear model/Multi-Layer-Perceptron(MLP) over node features



Label Propagation for Node Classification

- Collective Classification (Correct & Smooth)
  - **(2) Given a trained base predictor, we apply it to obtain soft labels for all the nodes.**
    - We expect these soft labels to be decently accurate.
    - **Can we use graph structure to post-process the predictions to make them more accurate?**

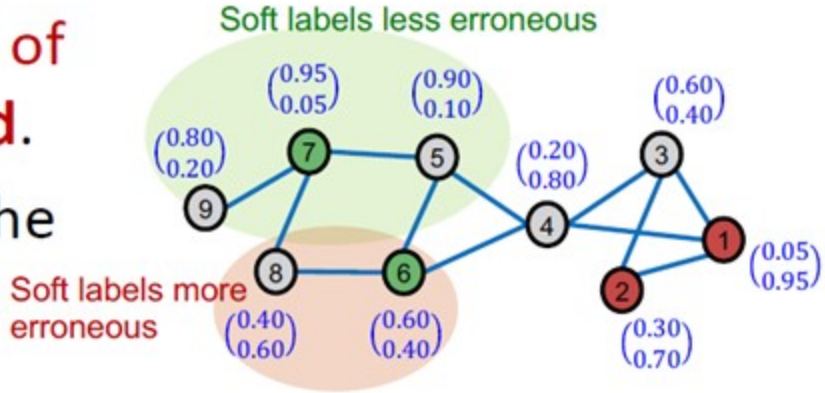


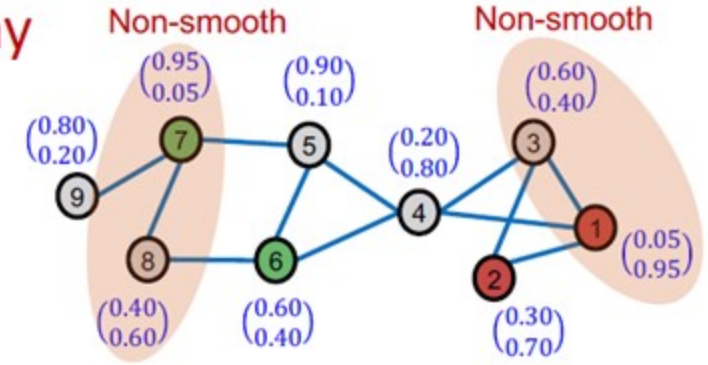
Label Propagation for Node Classification

- Collective Classification (Correct & Smooth)
  - **(3) C&S uses the 2-step procedure to post-process the soft predictions.**
    1. **Correct step**
    2. **Smooth step**
  - The key idea is that we expect errors in the base prediction to be positively correlated along edges in the graph.
    - In other words, an error at node u increases the chance of a similar error at neighbors of u.
    - Thus, we should "spread" such uncertainty over the graph.

Label Propagation for Node Classification

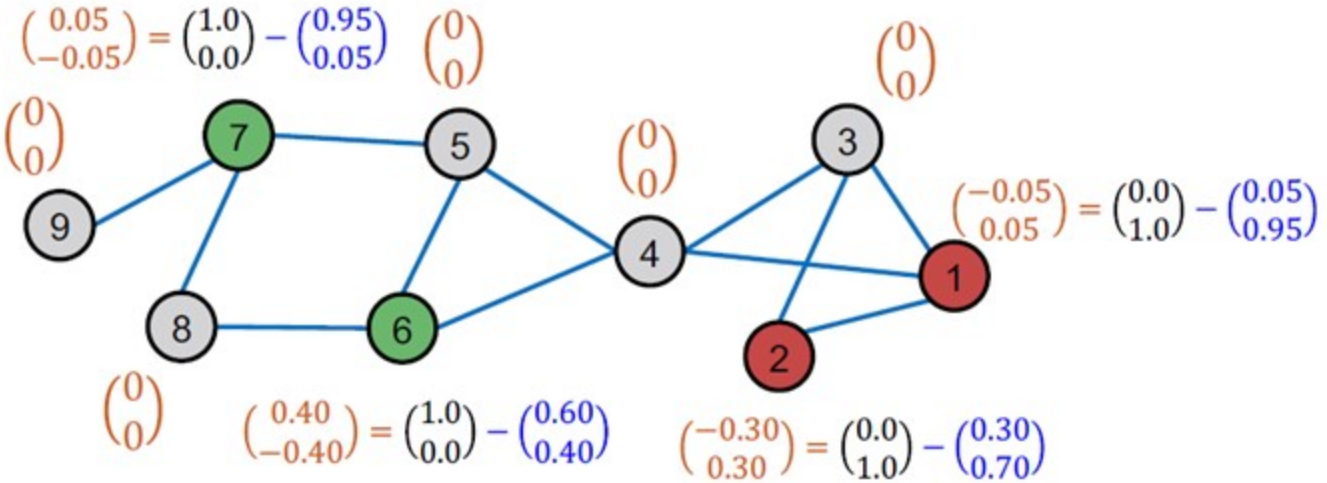
- Collective Classification (Correct & Smooth)
  - **Correct step**
    - The degree of the errors of the soft labels are **biased**.
    - We need to correct for the error bias.
  - **Smooth step**
    - The predicted soft labels may **not be smooth** over the graph.
    - We need to smoothen the soft labels.





Label Propagation for Node Classification

- Collective Classification (Correct & Smooth)
  - **Correct step:**
    - Compute training errors of nodes.
      - **Training error**: Ground-truth label minus soft label. Defined as 0 for unlabeled nodes.



Label Propagation for Node Classification

- Collective Classification (Correct & Smooth)
  - **Correct step (contd.):**
    - Diffuse training errors $\boldsymbol{E}^{(0)}$ along the edges.

Hyper-parameter

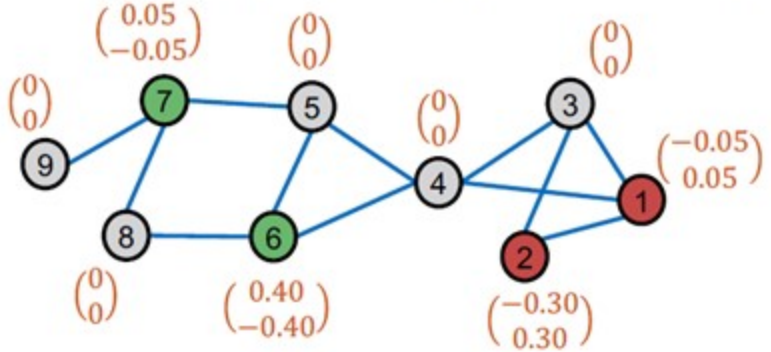
$$\boldsymbol{E}^{(t+1)} \leftarrow (1-\alpha)\cdot \boldsymbol{E}^{(t)} + \alpha \cdot \widetilde{\boldsymbol{A}}\boldsymbol{E}^{(t)}.$$

Diffuse training errors along the edges

Assumption: errors are similar for nearby nodes

Normalized diffusion matrix $\widetilde{\boldsymbol{A}} \equiv \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$

$\boldsymbol{D} \equiv \mathrm{Diag}(d_1, \ldots, d_N)$ be the degree matrix

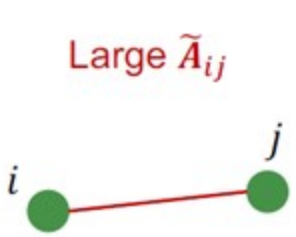


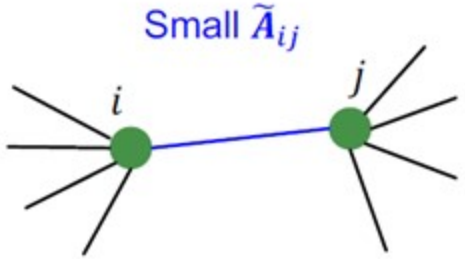
Initial training error matrix

$$\boldsymbol{E}^{(0)} = \begin{pmatrix} -0.05 & 0.05 \\ -0.30 & 0.30 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0.40 & -0.40 \\ 0.05 & -0.05 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$$

Label Propagation for Node Classification

- Collective Classification (Correct & Smooth)
  - **Correct step (contd.):**
    - If i and j are connected, the weight $\widetilde{\boldsymbol{A}}_{ij}$ is $\frac{1}{\sqrt{d_i}\sqrt{d_j}}$
    - **Intuition**:
      - **Large** if i and j are connected only with each other (no other nodes are connected to i and j).
      - **Small** if i and j are connected also connected with many other nodes.



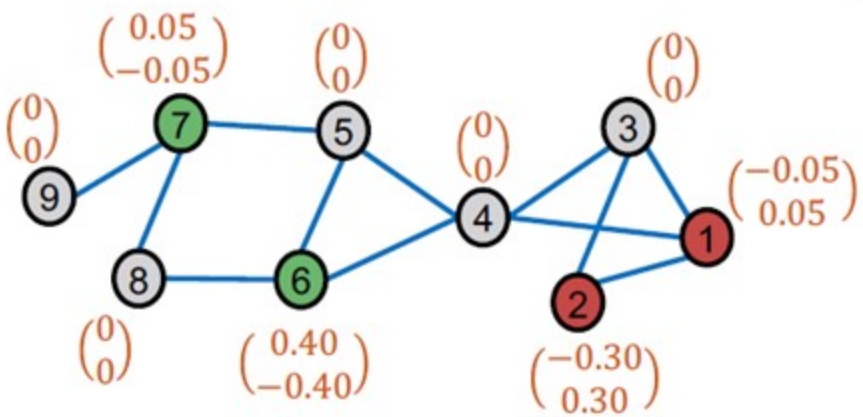


Label Propagation for Node Classification

- Collective Classification (Correct & Smooth)
  - **Correct step (contd.):**

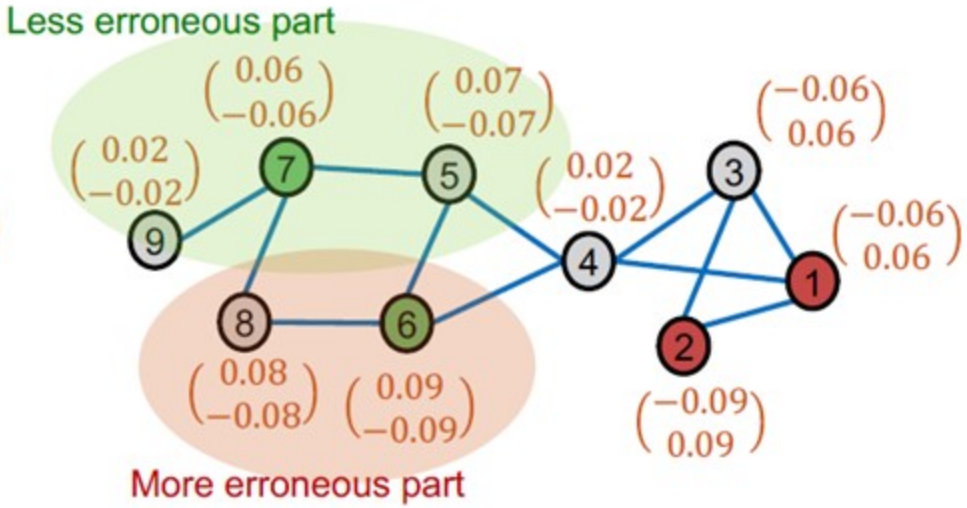
Before diffusion

$E^{(0)}$



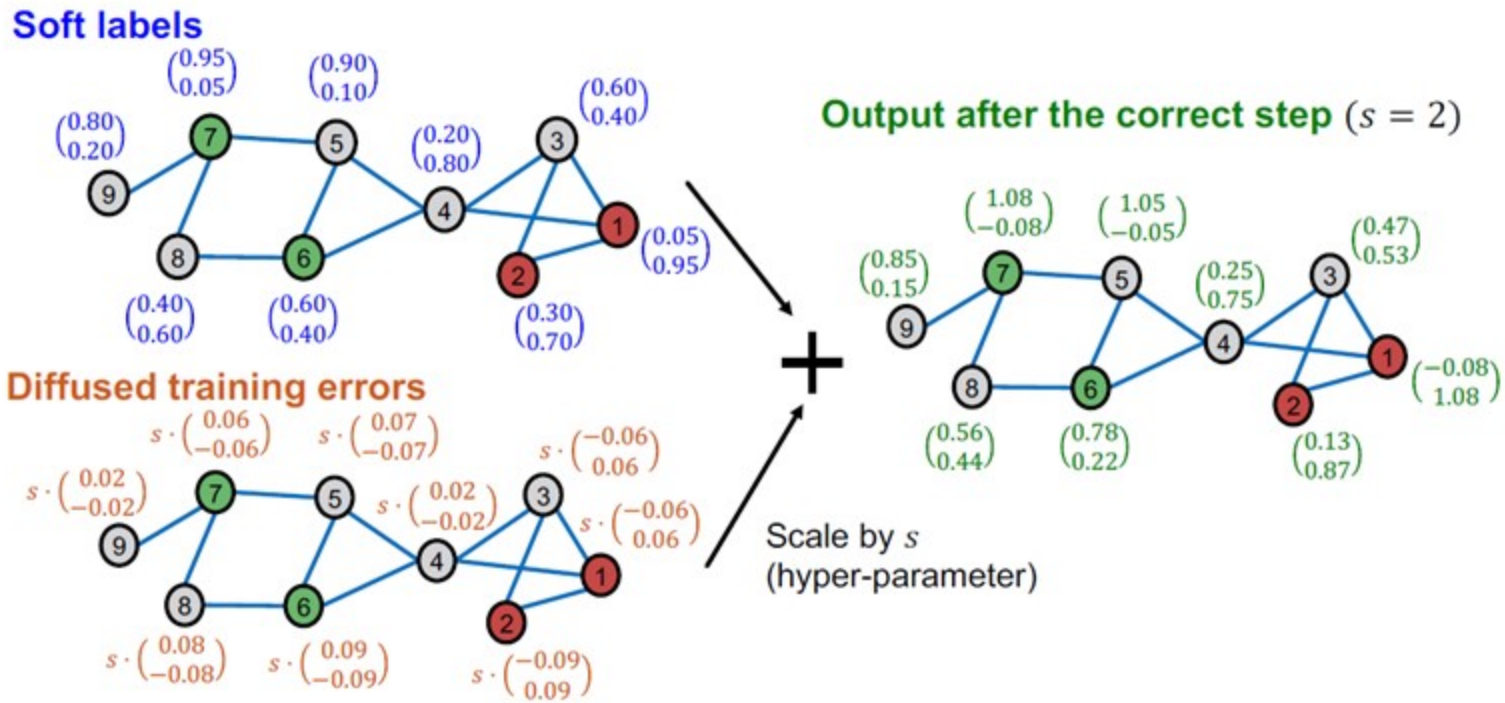
After diffusion

$E^{(3)}, \alpha = 0.8$



Label Propagation for Node Classification

- Collective Classification (Correct & Smooth)
  - **Correct step (contd.):**
    - **Add the scaled diffused training errors into the predicted soft labels**

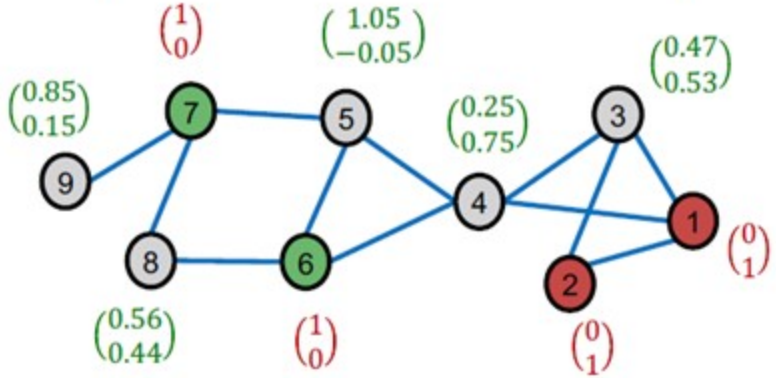


Label Propagation for Node Classification

- Collective Classification (Correct & Smooth)
  - **Smooth step:**

    Smoothen the corrected soft labels along the edges.
    - **Assumption**: Neighboring nodes tend to share the same labels.
    - **Note:** For training nodes, we use the **ground-truth hard labels** instead of the soft labels.

Input to the smooth step:

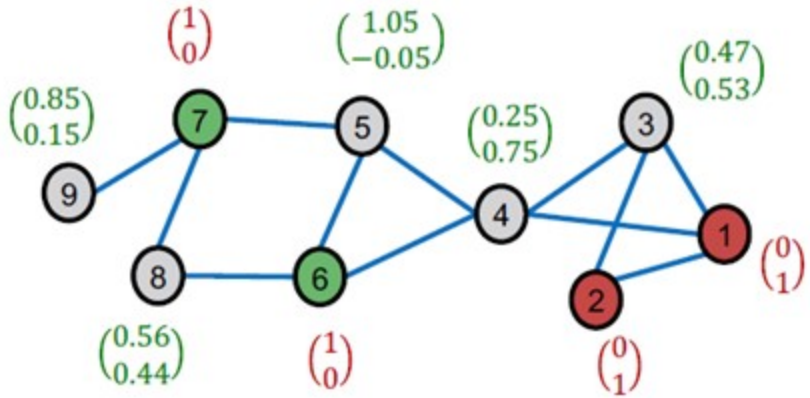


Label Propagation for Node Classification

- Collective Classification (Correct & Smooth)
  - **Smooth step:**
    - Diffuse label $\boldsymbol{Z}^{(0)}$ along the graph structure.
    - $Z^{(t+1)} \leftarrow (1-\alpha) \cdot Z^{(t)} + \alpha \cdot \tilde{\boldsymbol{A}} Z^{(t)}$

Hyper-parameter

Diffuse labels along the edges



Corrected label matrix

$$\boldsymbol{Z}^{(0)} = \begin{pmatrix} 0 & 1 \\ 0 & 1 \\ 0.47 & 0.53 \\ 0.25 & 0.75 \\ 1.05 & -0.05 \\ 1 & 0 \\ 1 & 0 \\ 0.56 & 0.44 \\ 0.85 & 0.15 \end{pmatrix}$$

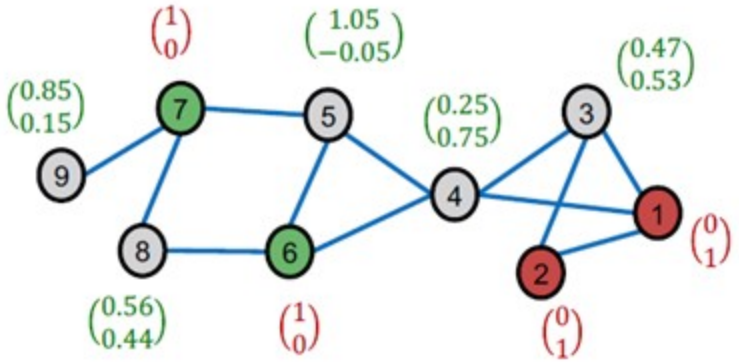
Label Propagation for Node Classification

- Collective Classification (Correct & Smooth)
  - **Smooth step:**

$$\boldsymbol{Z}^{(t+1)} \leftarrow (1-\alpha) \cdot \boldsymbol{Z}^{(t)} + \alpha \cdot \tilde{\boldsymbol{A}}\boldsymbol{Z}^{(t)}$$

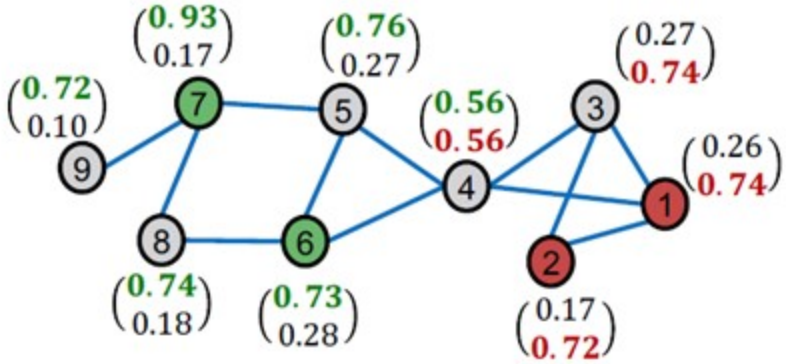
Before smoothing

$\boldsymbol{Z}^{(0)}$



After smoothing

$\boldsymbol{Z}^{(3)}, \alpha = 0.8$

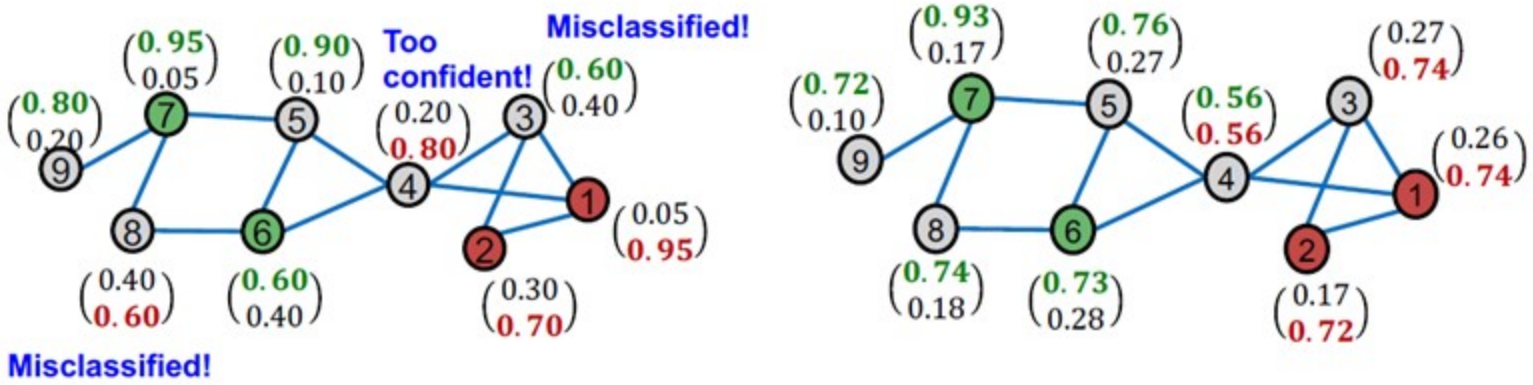


The final class prediction of C&S is the class with the maximum $Z^{(3)}$ score.

Note: The $\boldsymbol{Z}^{(3)}$ scores do not have direct probabilistic interpretation (e.g., not sum to 1 for each node), but larger scores indicate the classes are more likely.

Label Propagation for Node Classification

- Collective Classification (Correct & Smooth)
  - Example
    - Our toy example shows that C&S successfully improves base model performance using graph structure.



Label Propagation for Node Classification

- Collective Classification (Correct & Smooth)
  - Real-World Dataset
    - C&S significantly improves the performance of the base model (MLP).
    - C&S outperforms Smooth-only (no correct step) baseline.

Method	Classification accuracy (%) on ogbn-products dataset
MLP (base model)	63.41
MLP + smooth only	80.34
MLP + C&S	**84.18**