Label Propagation : A Practical Guide to Semi-Supervised Learning

Student Number: 23079448
GitHub Link: https://github.com/sb24ragnar/label_propagation_tutorial

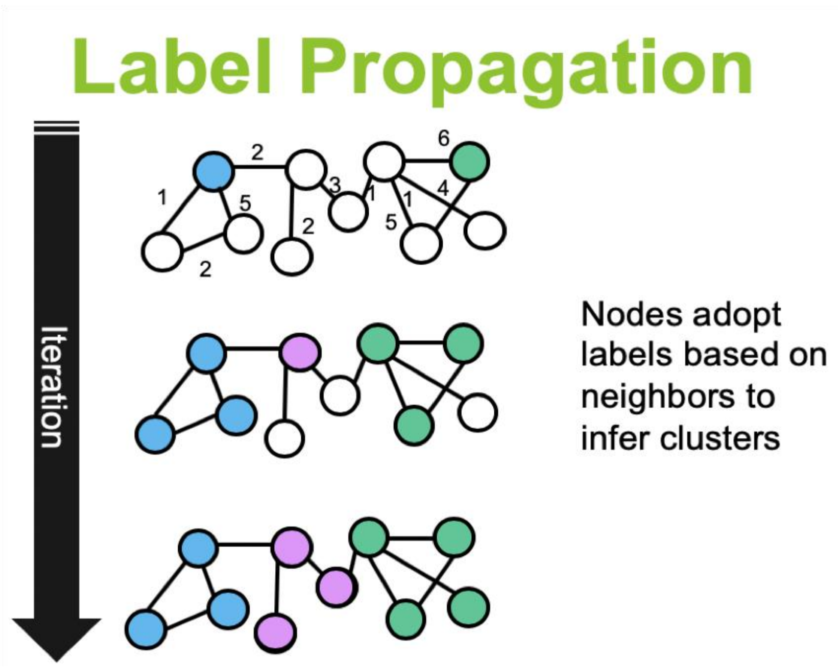
Introduction

Semi-supervised learning (SSL) is an area of machine learning that utilizes both labelled and unlabelled data to enhance model performance. Unlike supervised learning, which requires extensive labelled data, SSL effectively reduces labelling costs by making use of abundant unlabelled data. Label Propagation is a prominent SSL technique that propagates labels through a data similarity graph, assigning labels to unlabelled data based on labelled instances.

In today's data-driven world, machine learning models have become essential in tasks ranging from predicting customer preferences to detecting diseases. However, one significant challenge in building effective models is the cost and effort associated with labelling datasets. Manual labelling is not only expensive and time-consuming but also often requires specialized expertise. On the other hand, we usually have easy access to large volumes of unlabelled data, which traditional supervised learning models cannot fully leverage.

This is where **semi-supervised learning (SSL)** comes into picture combining a small amount of labelled data with a large amount of unlabelled data to build more accurate and robust models. SSL effectively bridges supervised and unsupervised learning methods, providing an efficient middle ground.

Among various SSL methods, **Label Propagation** stands out for its simplicity and effectiveness. Label Propagation intuitively **spreads known labels to unlabelled data points based on similarity**, much like how information or influence might naturally spread through a social network.



It's particularly powerful when labelled examples are few, making it ideal for real-world applications like medical diagnostics, text classification, and community detection in social networks.

In this tutorial, we'll explore Label Propagation in detail, breaking down its key concepts, practical implementation, strengths, limitations, and exciting real-world applications.

Dataset Overview: make_moons

The make_moons dataset is a synthetic two-dimensional dataset commonly used for testing and illustrating clustering, classification, and semi-supervised learning algorithms. It generates two interleaving half-circle shapes (hence "moons") that are not linearly separable, making it particularly useful for demonstrating nonlinear classification techniques.

It’s a simple toy dataset to visualize the clustering and classification algorithms.

Why Label Propagation Matters

The main idea behind Label Propagation is straightforward: similar data points are likely to share the same labels. Imagine trying to identify if an email is spam—similar emails tend to have similar labels. Label Propagation takes advantage of these natural patterns, helping us get the most out of limited labelled examples.

Key reasons why Label Propagation is attractive:

- **Efficiency:** It significantly reduces the need for manual labelling by smartly using unlabelled data.
- **Adaptability:** Works well across different types of data, including images, text, and structured databases.
- **Real-world Relevance:** Ideal for situations where labelling data extensively is impractical, like medical imaging or fraud detection.

By leveraging data's underlying structure, Label Propagation helps overcome challenges posed by limited labelled data, making it both practical and cost-effective.

How Label Propagation Works

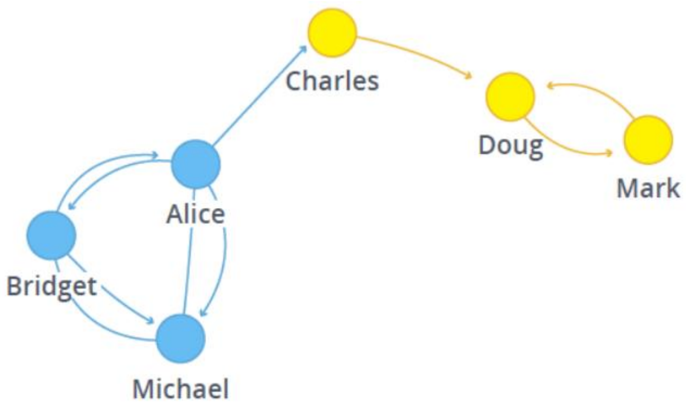
The Label Propagation process can be visualized as labelling nodes in a network. Here's how it works step-by-step:

1. Creating a Similarity Graph

We start by representing data points as nodes. Each node is connected to others based on similarity—much like social media connections between similar interests. These connections form a network where stronger similarities lead to stronger connections.

2. Assigning Initial Labels

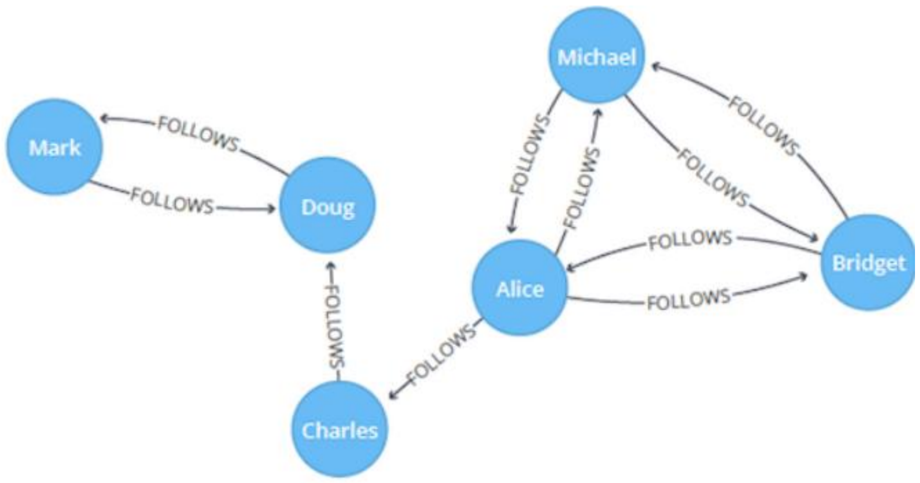
We initially label only a few nodes (data points), treating them as trusted sources. These labelled nodes become starting points from which labels propagate.



3. Spreading Labels Iteratively

In each iteration, labels from our trusted nodes spread out to neighbouring unlabelled nodes. Each unlabelled node looks at its neighbours and adopts the most common label, weighted by the strength of their connections. This iterative process continues until label assignments stabilize and no longer change significantly.

This approach is intuitive because it mirrors how opinions or information naturally spread through social groups. Over multiple rounds, the labels effectively "**fill in**" the unlabelled nodes, providing reliable predictions even with limited initial data.



Mathematics Behind Label Propagation

To precisely control label propagation, we use some straightforward mathematics:

Similarity Measures

We usually use measures like Euclidean distance or cosine similarity. For instance, the Gaussian similarity between two points x_i and x_j is calculated as:

$$W_{ij} = \exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right)$$

where x_i and x_j represent data points, and **σ** controls the width of the Gaussian kernel. The adjacency matrix W created from these similarities forms the basis for subsequent label propagation.

This value becomes the connection strength between points, influencing how labels propagate.

Propagation Algorithm

We then propagate labels iteratively using a simple update rule:

$$F^{(t+1)} = \alpha W F^{(t)} + (1 - \alpha) Y$$

Here:

- $F^{(t+1)}$Represents label distributions at iteration t.
- W is the similarity matrix (our network).
- Y represents our initial labels (fixed for labelled points).
- α determines how quickly labels propagate.

We keep updating labels until they stabilize, giving us consistent and reliable results.

Advanced Tips for Better Results

To achieve the best outcomes with Label Propagation, consider the following advanced tips:

Scalability

Use approximate nearest-neighbour algorithms or dimensionality reduction techniques like PCA to handle large datasets efficiently, making label propagation faster and more manageable.

Regularization

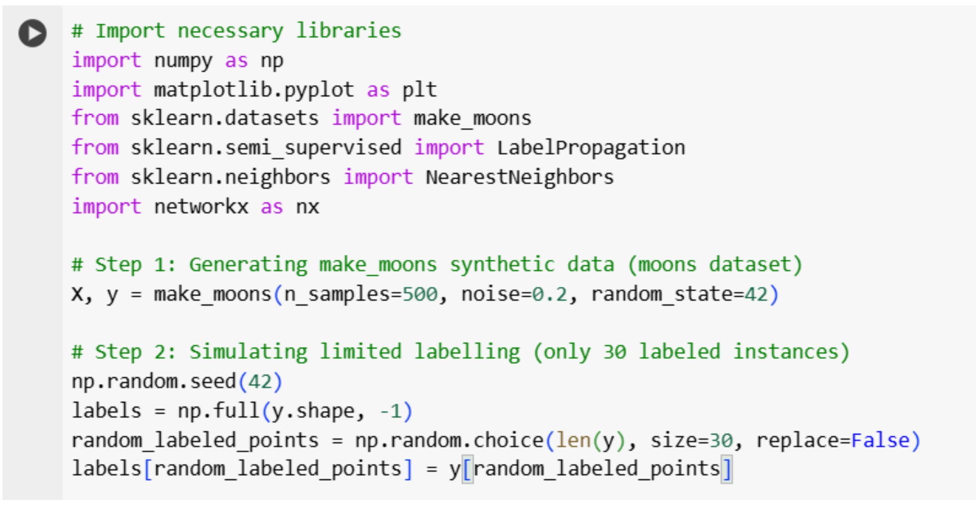
Employ regularization methods like label smoothing or controlled noise injection to reduce sensitivity to initial labelling errors and enhance robustness.

Parameter Tuning

Carefully tune hyperparameters like the propagation rate (α) and the number of neighbours to improve accuracy and reduce sensitivity to noise.

Practical Demonstration

Implementing Label Propagation with Python’s scikit-learn:

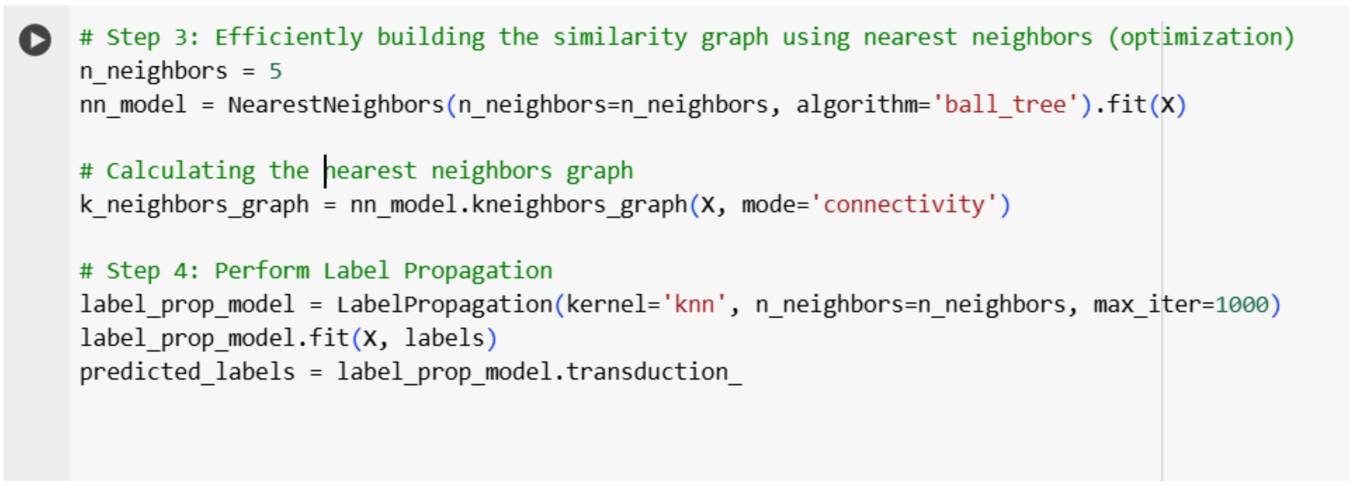


```python
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_moons
from sklearn.semi_supervised import LabelPropagation
from sklearn.neighbors import NearestNeighbors
import networkx as nx

# Step 1: Generating make_moons synthetic data (moons dataset)
X, y = make_moons(n_samples=500, noise=0.2, random_state=42)

# Step 2: Simulating limited labelling (only 30 labeled instances)
np.random.seed(42)
labels = np.full(y.shape, -1)
random_labeled_points = np.random.choice(len(y), size=30, replace=False)
labels[random_labeled_points] = y[random_labeled_points]
```

Created a synthetic, non-linearly separable dataset (moons dataset) to demonstrate the label propagation with less labelled and more unlabelled data and randomly selected 30 points out of n_samples points as labelled and the rest as unlabelled.



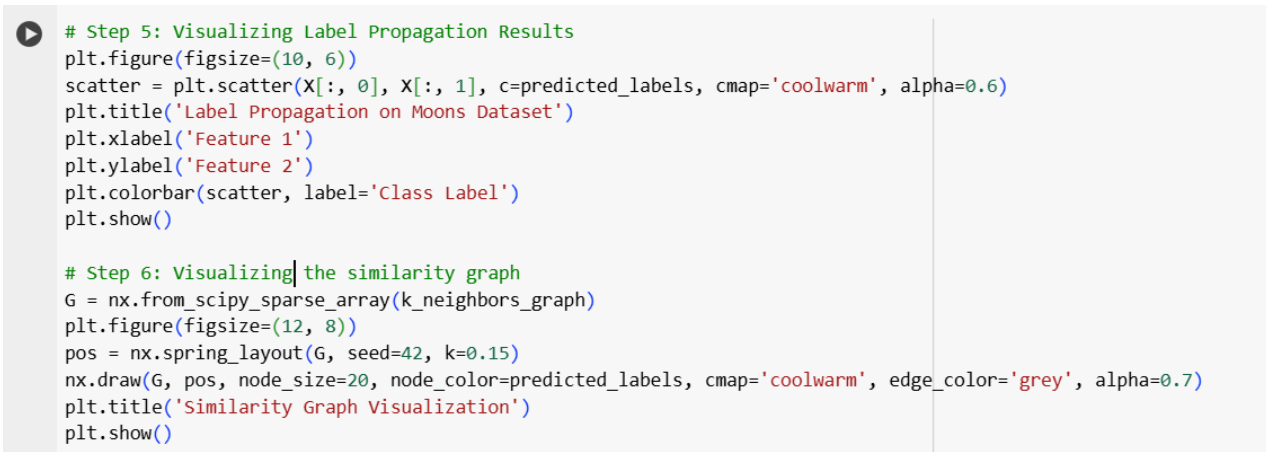
```python
# Step 3: Efficiently building the similarity graph using nearest neighbors (optimization)
n_neighbors = 5
nn_model = NearestNeighbors(n_neighbors=n_neighbors, algorithm='ball_tree').fit(X)

# Calculating the nearest neighbors graph
k_neighbors_graph = nn_model.kneighbors_graph(X, mode='connectivity')

# Step 4: Perform Label Propagation
label_prop_model = LabelPropagation(kernel='knn', n_neighbors=n_neighbors, max_iter=1000)
label_prop_model.fit(X, labels)
predicted_labels = label_prop_model.transduction_
```

Efficiently built a nearest-neighbour graph using the ball_tree algorithm, connecting each point to its closest neighbours, and implemented the LabelPropagation propagating known labels through the similarity graph, resulting in the classification of unlabelled points.

Visualizing the Label Propagation Result and Similarity Graph



```python
# Step 5: Visualizing Label Propagation Results
plt.figure(figsize=(10, 6))
scatter = plt.scatter(X[:, 0], X[:, 1], c=predicted_labels, cmap='coolwarm', alpha=0.6)
plt.title('Label Propagation on Moons Dataset')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.colorbar(scatter, label='Class Label')
plt.show()

# Step 6: Visualizing the similarity graph
G = nx.from_scipy_sparse_array(k_neighbors_graph)
plt.figure(figsize=(12, 8))
pos = nx.spring_layout(G, seed=42, k=0.15)
nx.draw(G, pos, node_size=20, node_color=predicted_labels, cmap='coolwarm', edge_color='grey', alpha=0.7)
plt.title('Similarity Graph Visualization')
plt.show()
```

Clearly visualized results, including the predicted labels and similarity graph structure gives a clear pictorial understanding and effectiveness of semi-supervised learning.

This simple demonstration showcases how effectively Label Propagation can label a dataset even with very few labelled examples, highlighting its usefulness in practical, real-world scenarios.

Advantages and Limitations

Advantages

- **Cost Reduction:** Significantly cuts down labelling costs, leveraging unlabelled data efficiently.
- **Simplicity & Interpretability:** The intuitive mechanism makes it easy to understand and explain results clearly.
- **Applicability:** Easily adaptable to various types of datasets and domains.

Limitations

- **Computational Costs:** Can be computationally intensive for very large datasets due to graph operations.
- **Dependence on Initial Labels:** The quality of initial labelling greatly influences overall accuracy.
- **Scalability Issues:** The iterative nature can slow down performance with extensive data.

Despite these limitations, the advantages typically outweigh the downsides, especially when used thoughtfully with optimal parameter tuning and proper data preprocessing.

Real-world Applications

Label Propagation is widely used across various domains due to its effectiveness with limited labelled data:

Machine Translation

It significantly improves translation accuracy, especially for languages where parallel labelled translations are scarce. By leveraging monolingual text data abundantly available, Label Propagation iteratively refines translation models, as demonstrated by research from Cheng et al. (2016).

Medical Imaging

Labelling medical scans (MRI, CT, X-rays) usually requires considerable expertise and is extremely costly. Label Propagation helps classify scans accurately from limited expert annotations, making diagnostics quicker and more reliable.

Spam Detection

In spam email detection, manual labelling of thousands of emails is impractical. Label Propagation helps classify emails based on similarity patterns, reducing labelling effort significantly and improving accuracy.

Social Network Analysis

Identifying communities and interest groups in social media networks is made easier through Label Propagation. With only limited initial labels (e.g., known user interests), the method effectively spreads these labels across the network, allowing for better user segmentation and community understanding.

Conclusion

Label Propagation provides an elegant and practical solution to leverage limited labelled data effectively. Its intuitive nature, combined with its applicability across diverse fields, makes it a valuable technique for any data scientist's toolkit. While challenges like computational complexity and sensitivity to initial labelling remain, thoughtful implementation and parameter tuning greatly mitigate these issues. Label Propagation stands as an exemplar of how semi-supervised learning can solve real-world problems efficiently, economically, and reliably.

References

1. Xiaojin Zhu and Zoubin Ghahramani (2002), Learning from Labeled and Unlabeled Data with Label Propagation, CMU-CALD-02-107

2. Semi-Supervised Learning—*O. Chapelle, B. Schölkopf, and A. Zien*, Eds. (London, U.K.: MIT Press, 2006, pp. 508, ISBN: 978-0-262-03358-9).

3. Label Propagation for Deep Semi-Supervised Learning, ***Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, Ondrej Chum***; Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 5070-5079

4. Beyer, K., Goldstein, J., Ramakrishnan, R., Shaft, U. (1999). When Is "Nearest Neighbor" Meaningful?

DOI : https://doi.org/10.1007/3-540-49257-7_15

5. Scikit-Learn Documentation , Label Propagation Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.semi_supervised.LabelPropagation.html