Here’s an overview of semi-supervised learning algorithms along with example code for each. We'll use the Iris dataset and a synthetic dataset to demonstrate these techniques.

Installation

Make sure you have the necessary libraries installed:

```
bash
Copy code
pip install numpy pandas scikit-learn matplotlib
```

1. Self-Training

In self-training, a model is first trained on a small amount of labeled data and then it iteratively labels the unlabeled data based on its predictions.

```
python
Copy code
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

# Load dataset
iris = load_iris()
X = iris.data
y = iris.target

# Split into labeled and unlabeled
X_labeled, X_unlabeled, y_labeled, _ = train_test_split(X, y, test_size=0.8,
random_state=42)
X_unlabeled = X_unlabeled[:40]  # Keep 40 for unlabeled

# Self-training model
model = RandomForestClassifier()
model.fit(X_labeled, y_labeled)

# Self-training process
for _ in range(10):  # 10 iterations
    # Predict on unlabeled data
    pseudo_labels = model.predict(X_unlabeled)

    # Add the predicted labels to the training set
    X_labeled = np.concatenate((X_labeled, X_unlabeled))
    y_labeled = np.concatenate((y_labeled, pseudo_labels))

    # Re-train the model
    model.fit(X_labeled, y_labeled)

# Test accuracy
_, X_test, y_test, _ = train_test_split(X, y, test_size=0.2, random_state=42)
predictions = model.predict(X_test)
print('Self-Training Accuracy:', accuracy_score(y_test, predictions))
```

2. Co-Training

In co-training, two different models are trained on the same labeled data, and they help each other label the unlabeled data.

```
python
Copy code
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Load dataset
iris = load_iris()
X = iris.data
y = iris.target

# Split into labeled and unlabeled
X_labeled, X_unlabeled, y_labeled, _ = train_test_split(X, y, test_size=0.8,
random_state=42)
X_unlabeled = X_unlabeled[:40]

# Train two models
model1 = RandomForestClassifier()
model2 = LogisticRegression(max_iter=200)

# Initial training on labeled data
model1.fit(X_labeled, y_labeled)
model2.fit(X_labeled, y_labeled)

# Co-training process
for _ in range(10):
    # Predictions by both models
    pseudo_labels1 = model1.predict(X_unlabeled)
    pseudo_labels2 = model2.predict(X_unlabeled)

    # Select confident predictions
    confident_indices1 = np.where(np.max(model1.predict_proba(X_unlabeled),
axis=1) > 0.7)[0]
    confident_indices2 = np.where(np.max(model2.predict_proba(X_unlabeled),
axis=1) > 0.7)[0]

    # Add confident predictions to the labeled set
    if confident_indices1.size > 0:
        X_labeled = np.concatenate((X_labeled, X_unlabeled[confident_indices1]))
        y_labeled = np.concatenate((y_labeled,
pseudo_labels1[confident_indices1]))

    if confident_indices2.size > 0:
        X_labeled = np.concatenate((X_labeled, X_unlabeled[confident_indices2]))
        y_labeled = np.concatenate((y_labeled,
pseudo_labels2[confident_indices2]))

    # Remove labeled instances from unlabeled data
    X_unlabeled = np.delete(X_unlabeled, np.concatenate((confident_indices1,
confident_indices2)), axis=0)

    # Re-train models
    model1.fit(X_labeled, y_labeled)
    model2.fit(X_labeled, y_labeled)

# Test accuracy
_, X_test, y_test, _ = train_test_split(X, y, test_size=0.2, random_state=42)
predictions = model1.predict(X_test)
print('Co-Training Accuracy (Model 1):', accuracy_score(y_test, predictions))
```

3. Graph-Based Semi-Supervised Learning

Graph-based methods use a graph to represent the relationships between instances, where nodes are instances and edges represent their similarity. For simplicity, we'll use the `sklearn` implementation of Label Spreading.

```
python
Copy code
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.semi_supervised import LabelSpreading
from sklearn.metrics import accuracy_score

# Load dataset
iris = load_iris()
X = iris.data
y = iris.target

# Create unlabeled data by masking
y_unlabeled = np.copy(y)
y_unlabeled[:100] = -1  # Mask the first 100 labels as unlabeled

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y_unlabeled,
test_size=0.2, random_state=42)

# Graph-based semi-supervised learning
model = LabelSpreading()
model.fit(X_train, y_train)

# Predictions
predictions = model.predict(X_test)
print('Graph-Based Semi-Supervised Learning Accuracy:',
accuracy_score(y_test[y_test != -1], predictions[y_test != -1]))
```

Summary

This code demonstrates three semi-supervised learning algorithms: Self-Training, Co-Training, and Graph-Based methods. You can adapt the datasets and parameters according to your specific needs. If you have any questions or need further details, feel free to ask!