【機器學習基礎】半監督算法概覽(Python)

機器學習初學者 今天

以下文章來源於算法進階， 作者算法進階

算法進階

AI碼農一枚，熱愛分享Python、機器學習算法等原創好文和項目。歡迎一起學習和交流～

前言

前階段時間梳理了機器學習開發實戰的系列文章：

1、Python機器學習入門指南（全）

2、Python數據分析指南(全)

3、一文歸納Ai數據增強之法

4、一文歸納Python特徵生成方法(全)

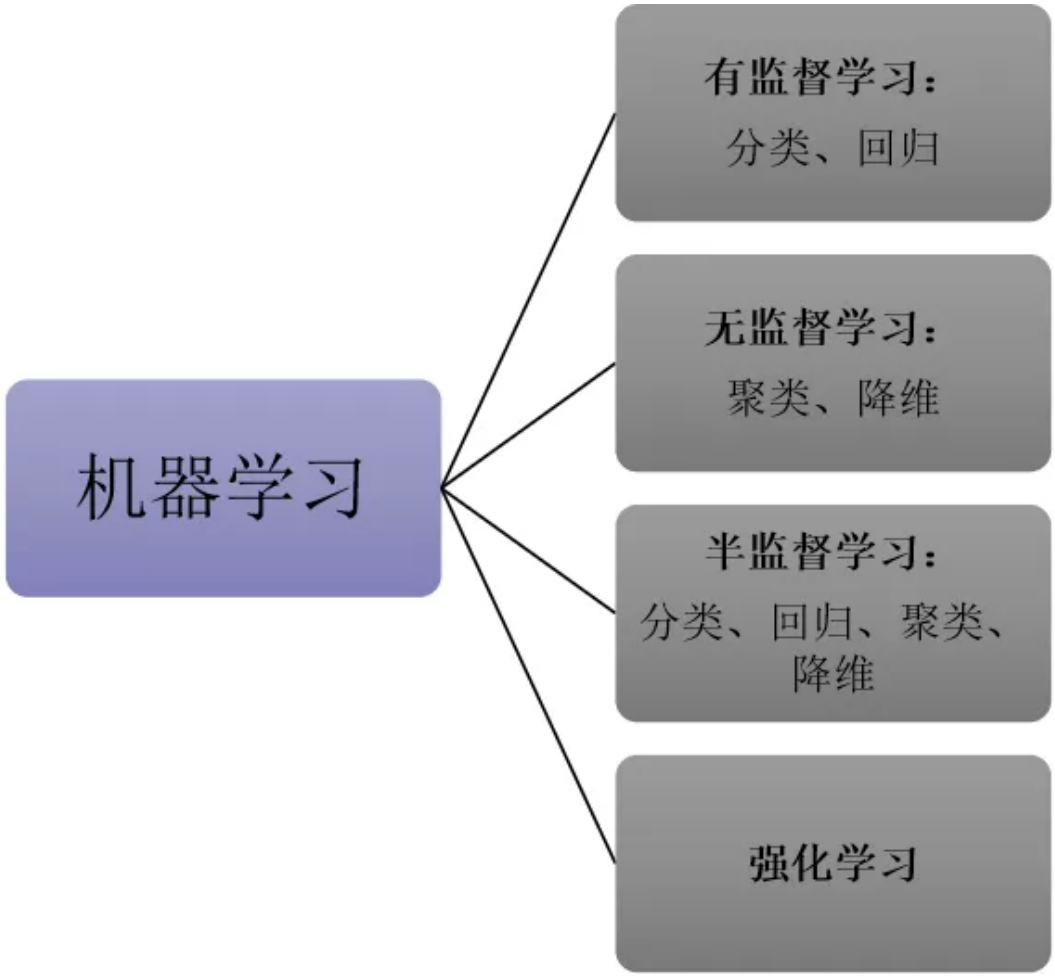
5、Python特徵選擇(全)

6、一文歸納Ai調參煉丹之法

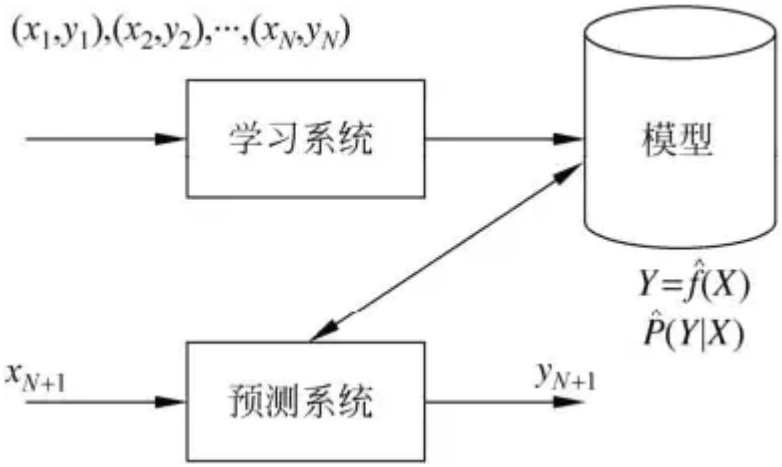
現階段的寫作計劃會對各類機器學習算法做一系列的原理概述及實踐，主要包括無監督聚類、異常檢測、**半監督算法**、強化學習、集成學習等。

一、機器學習簡介

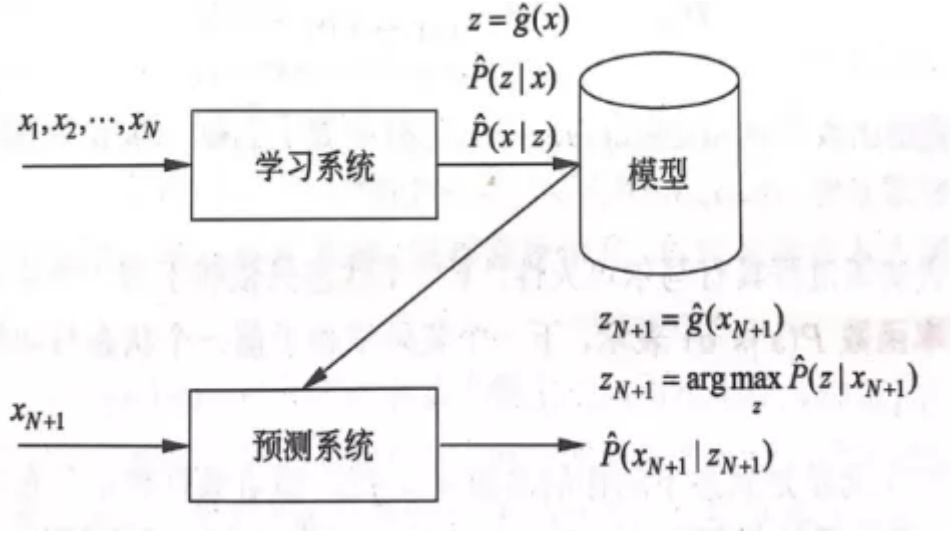
機器學習按照數據的標籤情況可以細分為：監督學習，無監督學習，半监督學習以及強化學習。



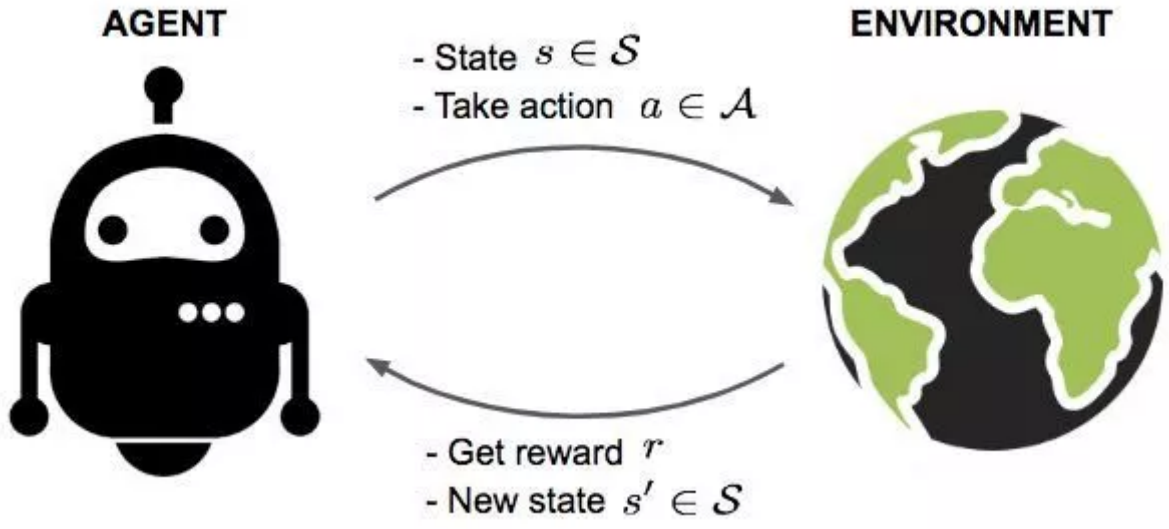
- 監督學習是利用數據特徵及其標籤D = { (x 1 , y 1) , … , (x l , y l) } 學習輸入到輸出的映射 f : X → Y 的方法。



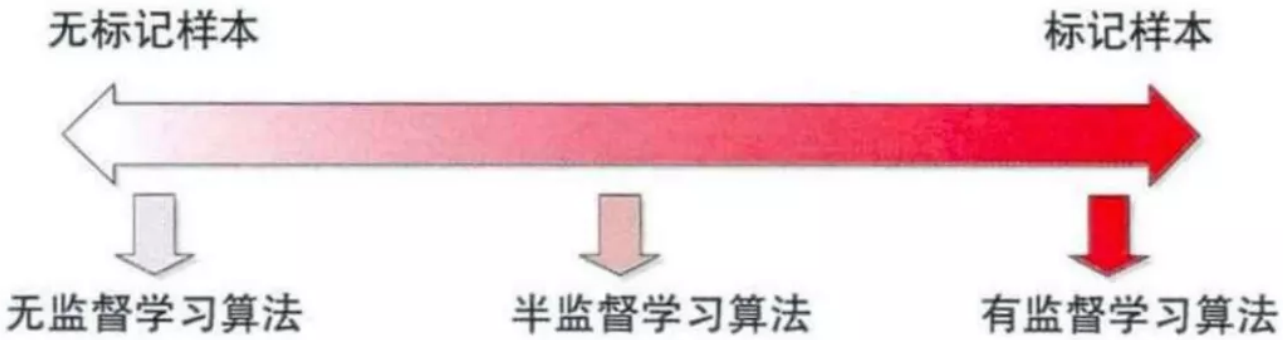
- 無監督學習是僅利用無類標籤的樣本數據特徵D = { x 1 ，… ， x n } 學習其對應的簇標籤、特徵表示等方法。



- 強化學習從某種程度可以看作是有延遲標籤信息的監督學習。



- 半監督學習是介於傳統監督學習和無監督學習之間，其思想是在標記樣本數量較少的情況下，通過在模型訓練中直接引入無標記樣本，以充分捕捉數據整體潛在分佈，以改善如傳統無監督學習過程盲目性、監督學習在訓練樣本不足導致的學習效果不佳的問題。



> 半监督学习的有效性通常基于如下假设：1）平滑假设：稠密数据区域的两个距离很近的样例的类标签相似。2）聚类假设：当两个样例位于同一聚类簇时，很大的概率下有相同的类标签。3）流形假设：高维数据嵌入到低维流形中，当两个样例位于低维流形中的一个小局

> 部邻域内时，具有相似的类标签。当模型假设不正确时，无标签的样本可能无法有效地提供增益信息，反而会恶化学习性能。

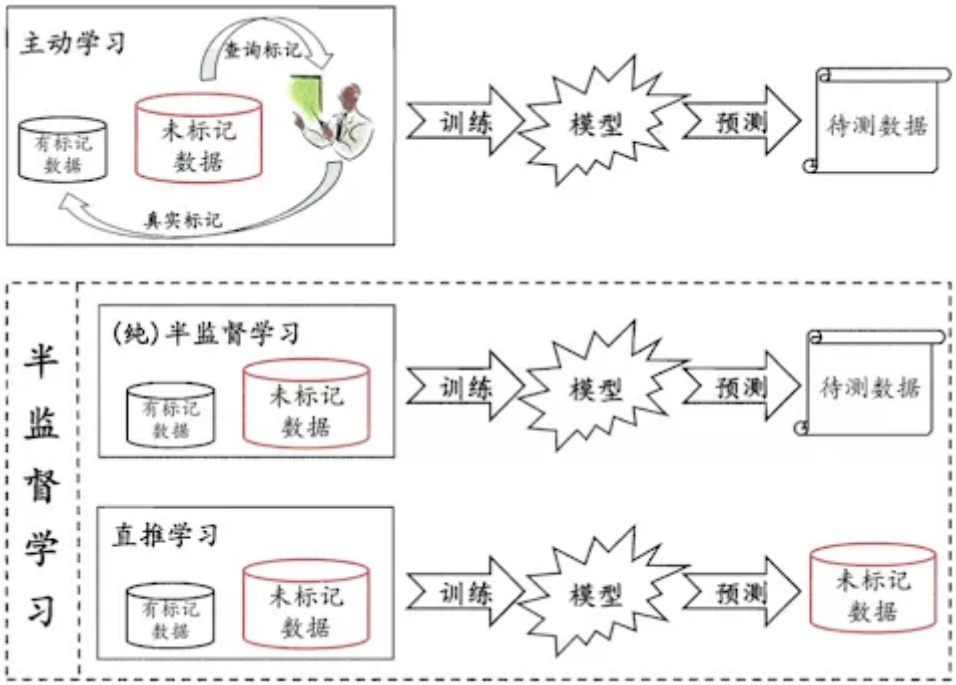
二、半监督算法的类别

2.1 按理论差异划分

按照统计学习理论差异，半监督学习可以分为：(纯)归纳半监督学习和直推学习。

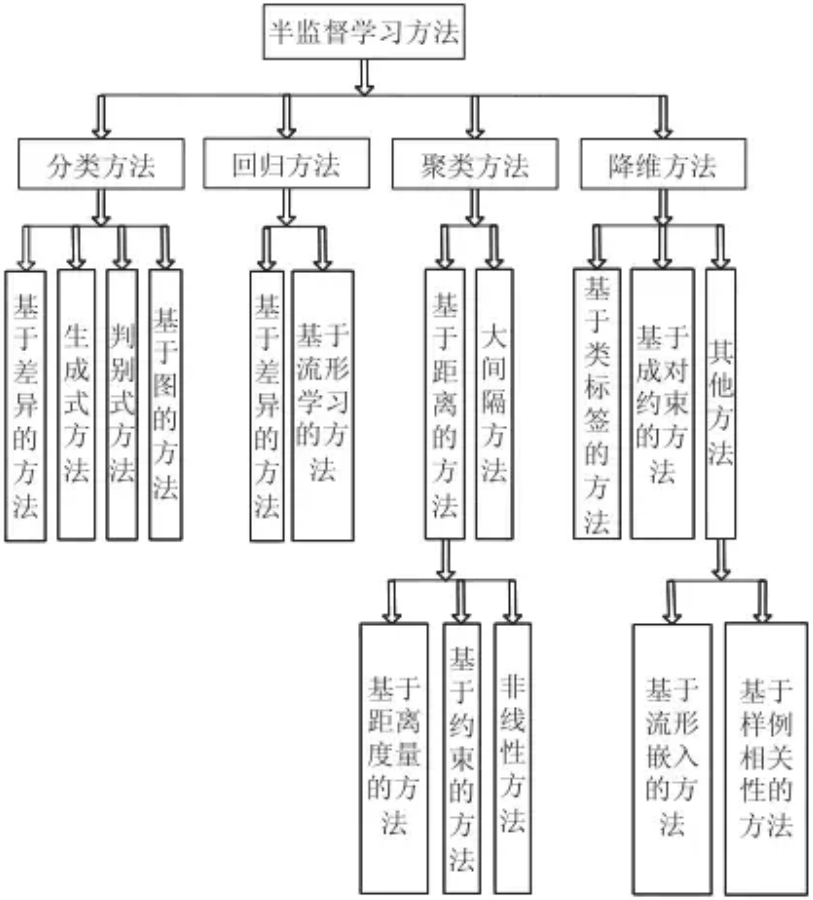
直推学习只处理样本空间内给定的训练数据，利用训练数据中有类标签的样本和无类标签的样例进行训练，仅预测训练数据中无类标签的样例的类标签，典型如标签传播算法(LPA)。

归纳半监督学习处理整个样本空间中所有给定和未知的样例，不仅预测训练数据中无类标签的样例的类标签，更主要的是预测未知的测试样例的类标签，典型如半监督SVM。



2.2 按学习场景划分

从不同的学习场景看，半监督学习可分为四类：半监督分类（Semi-supervised classification）、半监督回归（Semi-supervised regression）、半监督聚类（Semi-supervised clustering）及半监督降维（Semi-supervised dimensionality reduction）。



- 半监督分类 半监督分类算法的思想是通过大量的未标记样本帮助学习一个好的分类系统，代表算法可以划分为四类，包括生成式方法、判别式方法、半监督图算法和基于差异的半监督方法(此外还可扩展出半监督深度学习方法，限于篇幅本文没有展开)。结合现实情况多数为半监督分类场景，下节会针对半监督分类算法原理及实战进行展开。

- 半监督聚类 半监督聚类算法的思想是如何利用先验信息以更好地指导未标记样本的划分过程。现有的算法多数是在传统聚类算法基础上引入监督信息发展而来，基于不同的聚类算法可以将其扩展成不同的半监督聚类算法。

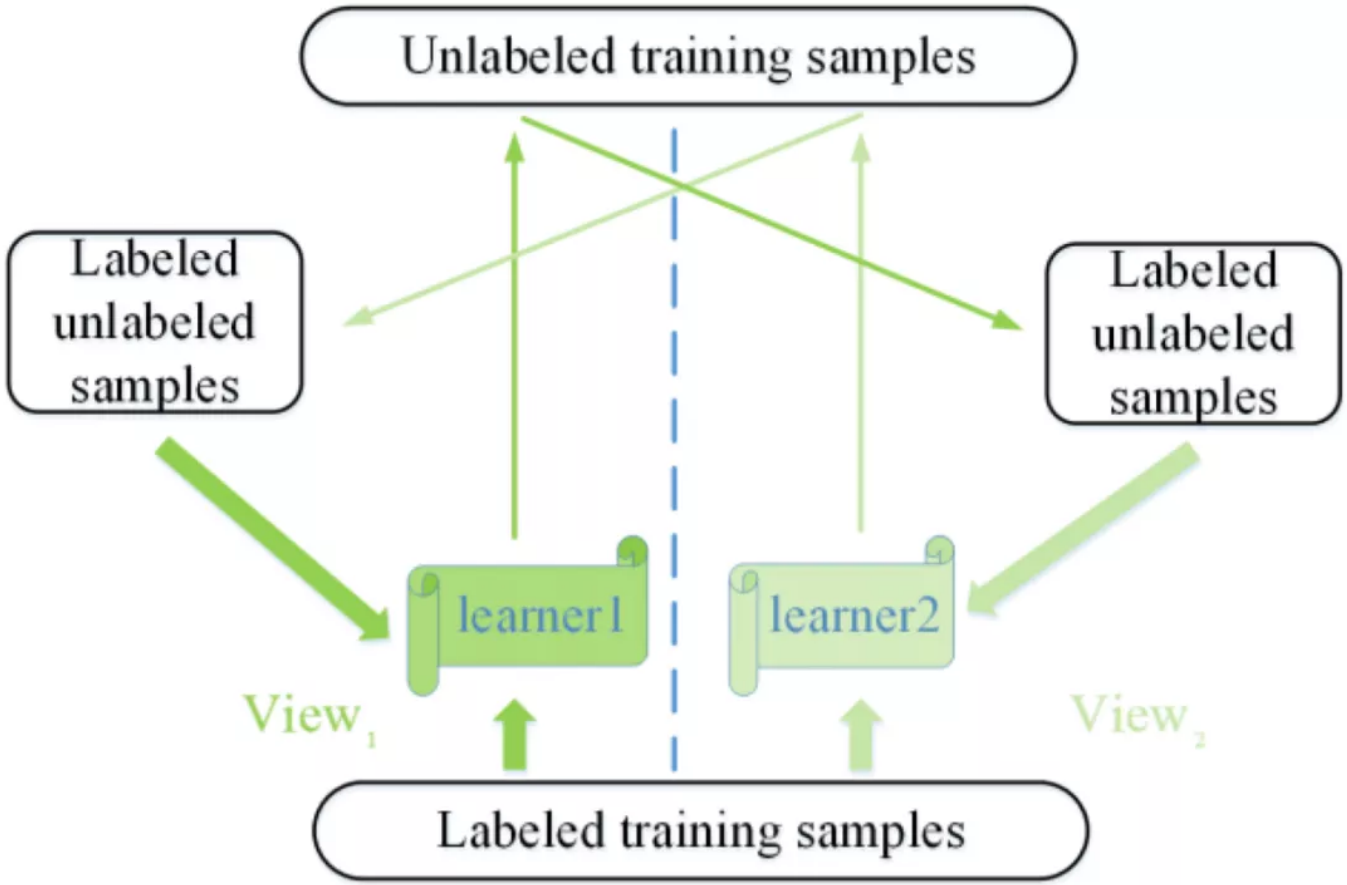
- 半监督回归 半监督回归算法的思想是通过引入大量的未标记样本改进监督学习方法的性能，训练得到性能更优的回归器。现有的方法可以归纳为基于协同训练(差异)的半监督回归和基于流形的半监督回归两类。

- 半监督降维 半监督降维算法的思想在大量的无类标签的样例中引入少量的有类标签的样本，利用监督信息找到高维数据的低维结构表示，同时保持数据的内在固有信息。而利用的监督信息既可以是样例的类标签，也可以是成对约束信息，还可以是其他形式的监督信息。主要的半监督降维方法有基于类标签的方法、基于成对约束等方法。

三、半监督分类算法(Python)

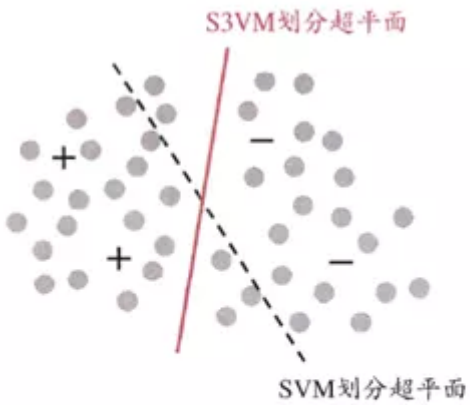
3.1 基于差异的方法

基于差异的半监督学习起源于协同训练算法，其思想是利用多个拟合良好的学习器之间的差异性提高泛化能力。假设每个样本可以从不同的角度（view）训练出不同的分类器，然后用这些从不同角度训练出来的分类器对无标签样本进行分类，再选出认为可信的无标签样本加入训练集中。



3.2 判别式方法

判别式方法利用最大间隔算法同时训练有类标签的样本和无类标签的样例学习决策边界，使其通过低密度数据区域，并且使学习得到的分类超平面到最近的样例的距离间隔最大。常见的如直推式支持向量机（TSVM）及最近邻（KNN）等。



半监督支持向量机与低密度分隔（“+” “−”分别表示有标记的正、反例，灰色点表示未标记样本）

TSVM采用局部搜索的策略来进行迭代求解，即首先使用有标记样本集训练出一个初始SVM，接着使用该学习器对未标记样本进行打标，这样所有样本都有了标记，并基于这些有标记的样本重新训练SVM，之后再寻找易出错样本不断调整。

```
import random
import numpy as np
import sklearn.svm as svm
from sklearn.datasets import make_classification


class TSVM(object):
    '''
    半监督TSVM
    '''
    def __init__(self, kernel='linear'):
        self.Cl, self.Cu = 1.5, 0.001
```

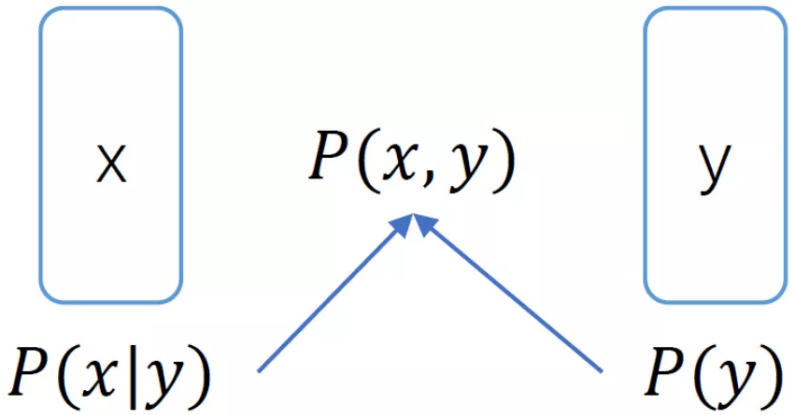
```
        self.kernel = kernel
        self.clf = svm.SVC(C=1.5, kernel=self.kernel)

    def train(self, X1, Y1, X2):
        N = len(X1) + len(X2)
        # 样本权值初始化
        sample_weight = np.ones(N)
        sample_weight[len(X1):] = self.Cu

        # 用已标注部分训练出一个初始SVM
        self.clf.fit(X1, Y1)

        # 对未标记样本进行标记
        Y2 = self.clf.predict(X2)
        Y2 = Y2.reshape(-1,1)

        X = np.vstack([X1, X2])
        Y = np.vstack([Y1, Y2])

        # 未标记样本的序号
        Y2_id = np.arange(len(X2))

        while self.Cu < self.Cl:
            # 重新训练SVM, 之后再寻找易出错样本不断调整
            self.clf.fit(X, Y, sample_weight=sample_weight)
            while True:
                Y2_decision = self.clf.decision_function(X2)    # 参数实例到决策超平面的距离
                Y2 = Y2.reshape(-1)
                epsilon = 1 - Y2 * Y2_decision
                negative_max_id = Y2_id[epsilon==min(epsilon)]
                # print(epsilon[negative_max_id][0])
                if epsilon[negative_max_id][0] > 0:
```

```
                if epsilon[negative_max_id][0] > 0:
                    # 寻找很可能错误的未标记样本，改变它的标记成其他标记
                    pool = list(set(np.unique(Y1))-set(Y2[negative_max_id]))
                    Y2[negative_max_id] = random.choice(pool)
                    Y2 = Y2.reshape(-1, 1)
                    Y = np.vstack([Y1, Y2])

                    self.clf.fit(X, Y, sample_weight=sample_weight)
                else:
                    break
            self.Cu = min(2*self.Cu, self.Cl)
            sample_weight[len(X1):] = self.Cu

    def score(self, X, Y):
        return self.clf.score(X, Y)

    def predict(self, X):
        return self.clf.predict(X)

if __name__ == '__main__':
    features, labels = make_classification(n_samples=200, n_features=3,
                                           n_redundant=1, n_repeated=0,
                                           n_informative=2, n_clusters_per_class=2)
    n_given = 70
    # 取前n_given个数字作为标注集
    X1 = np.copy(features)[:n_given]
    X2 = np.copy(features)[n_given:]
    Y1 = np.array(np.copy(labels)[:n_given]).reshape(-1,1)
    Y2_labeled = np.array(np.copy(labels)[n_given:]).reshape(-1,1)
    model = TSVM()
    model.train(X1, Y1, X2)
```

```
accuracy = model.score(X2, Y2_labeled)
print(accuracy)
```

3.3 生成式方法

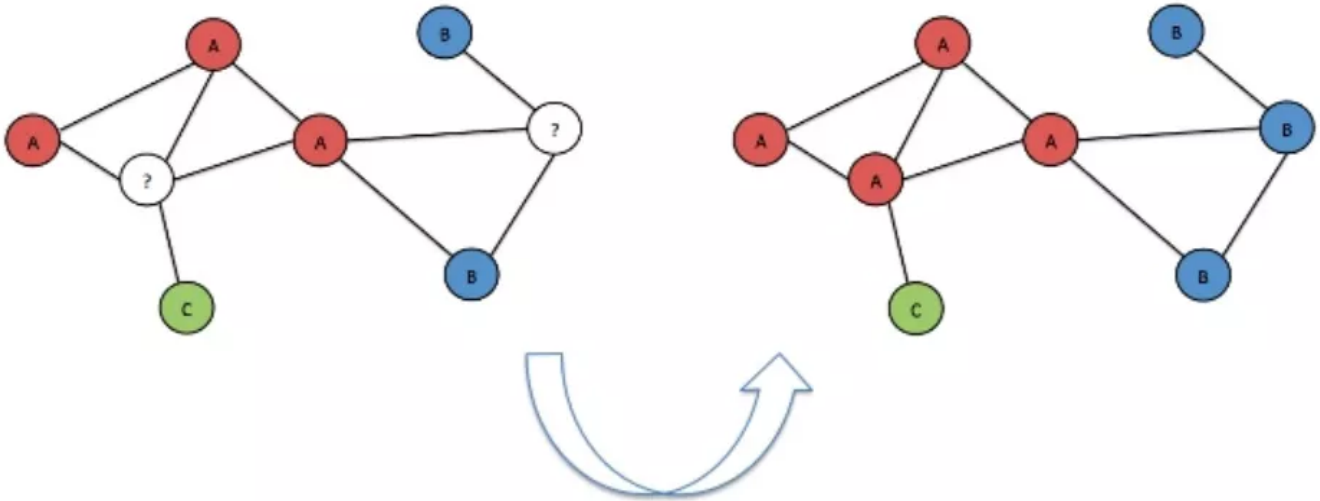
生成式的模型有高斯模型、贝叶斯网络、朴素贝叶斯、隐马尔可夫模型等，方法关键在于对来自各个种类的样本分布进行假设以及对所假设模型的参数估计。首先通过假设已知样本数据的密度函数 p(x|yi)的形式，比如多项式、高斯分布等。接着可采用迭代算法(如 EM 算法)计算 p(x|yi)的参数，然后根据贝叶斯全概率公式对全部未标签样本数据进行分类。



生成式方法可以直接关注半监督学习和决策中的条件概率问题，避免对边缘概率或联合概率的建模以及求解，然而该方法对一些假设条件比较苛刻，一旦假设的 p(x|yi)与样本数据的实际分布情况差距比较大，其分类效果往往不佳。

3.4 基于图半监督学习方法

基于图的方法的实质是标签传播，基于流形假设根据样例之间的几何结构构造边（边的权值可以用样本间的相近程度），用图的结点表示样例，利用图上的邻接关系将类标签从有类标签的样本向无类标签的样例传播。基于图的方法通常图计算复杂度较高，且对异常图结构缺乏鲁棒性，主要方法有最小分割方法、标签传播算法（LPA）和流形方法 (manifold method)等。



标签传播算法（LPA）是基于图的半监督学习算法，基本思路是从已标记的节点标签信息来预测未标记的节点标签信息。

1、首先利用样本间的关系（可以是样本客观关系，或者利用相似度函数计算样本间的关系）建立完全图模型。

2、接着向图中加入已标记的标签信息，无标签节点是在用一个唯一的标签初始化。

3、该算法会重复地将一个节点的标签设置为该节点的相邻节点中出现频率最高(有权图需要考虑权重)的标签，重复迭代，直到标签不变算法收敛。

```
import random
import networkx as nx
import matplotlib.pyplot as plt

class LPA():
    '''
    标签传播算法：传播标签来划分社区
    算法终止条件：迭代次数超过设定值
    self.G：图
```

```
    return：None
    '''
    def __init__(self, G, iters=10):
        self.iters = iters
        self.G = G

    def train(self):
        max_iter_num = 0 # 迭代次数

        while max_iter_num < self.iters:
            max_iter_num += 1
            print('迭代次数',max_iter_num)

            for node in self.G:
                count = {} # 记录邻居节点及其标签
                for nbr in self.G.neighbors(node): # node的邻居节点
                    label = self.G.node[nbr]['labels']
                    count[label] = count.setdefault(label,0) + 1

                # 找到出现次数最多的标签
                count_items = sorted(count.items(),key=lambda x:-x[-1])
                best_labels = [k for k,v in count_items if v == count_items[0][1]]
                # 当多个标签频次相同时随机选取一个标签
                label = random.sample(best_labels,1)[0]
                self.G.node[node]['labels'] = label # 更新标签

    def draw_picture(self):
        # 画图
        node_color = [float(self.G.node[v]['labels']) for v in self.G]
        pos = nx.spring_layout(self.G) # 节点的布局为spring型
        plt.figure(figsize = (8,6)) # 图片大小
```

```
        nx.draw_networkx(self.G,pos=pos,node_color=node_color)
        plt.show()

if __name__ == "__main__":
    G = nx.karate_club_graph() # 空手道
    # 给节点添加标签
    for node in G:
        G.add_node(node, labels = node) # 用labels的状态
    model = LPA(G)
    # 原始节点标签
    model.draw_picture()
    model.train()
    com = set([G.node[node]['labels'] for node in G])
    print('社区数量',len(com))
    # LPA节点标签
    model.draw_picture()
```

文章首发于算法进阶，公众号阅读原文可访问GitHub项目源码

往期精彩回顾

- 适合初学者入门人工智能的路线及资料下载
- 机器学习及深度学习笔记等资料打印
- 机器学习在线手册
- 深度学习笔记专辑
- 《统计学习方法》的代码复现专辑
- AI基础下载
- 机器学习的数学基础专辑
- 温州大学《机器学习课程》视频

本站qq群851320808，加入微信群请扫码：

阅读原文

喜欢此内容的人还喜欢

灵魂拷问：机器学习、深度学习专业已经沦为调包专业了吗?

機器學習初學者

雲南普洱邊境管理支隊連續9年繳毒量破噸!

國家移民管理局

廈大才女遠嫁印度當小老婆，為了愛情改國籍改姓氏，12年後卻暴露了真面目!

LULU畫報