Table Of Contents：

今日工作

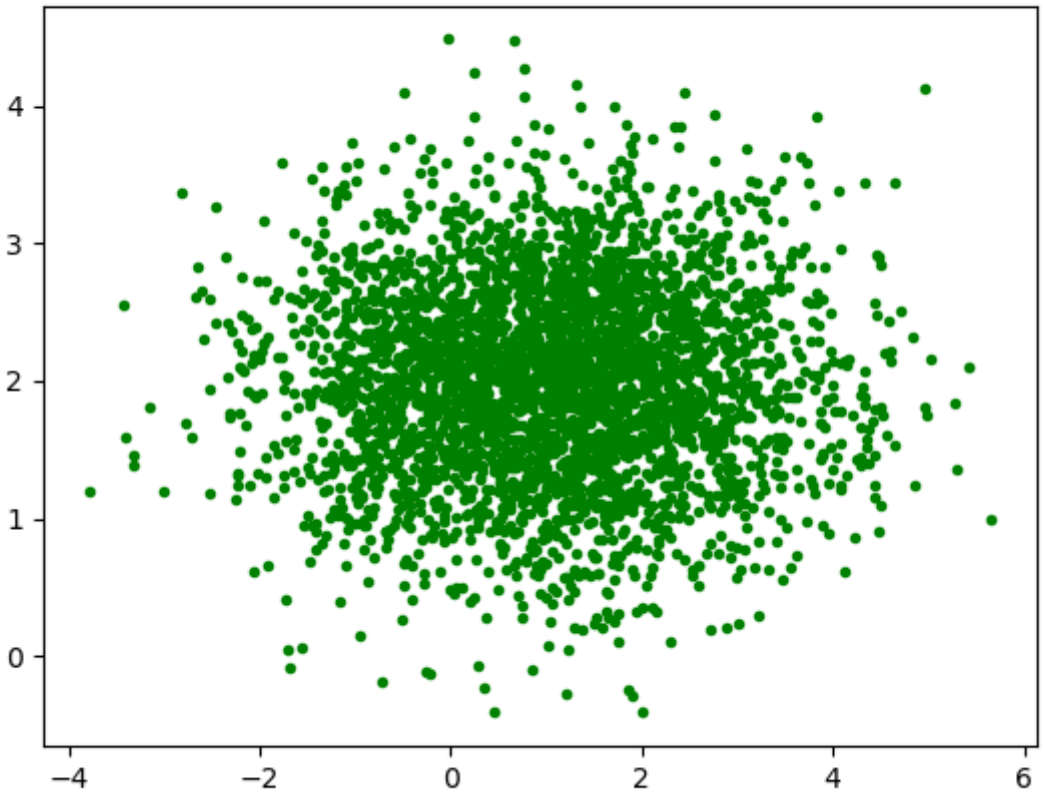
1、回顾《计算机视觉 模型、学习和推理》第七章。

2、编程实现混合高斯模型的拟合。

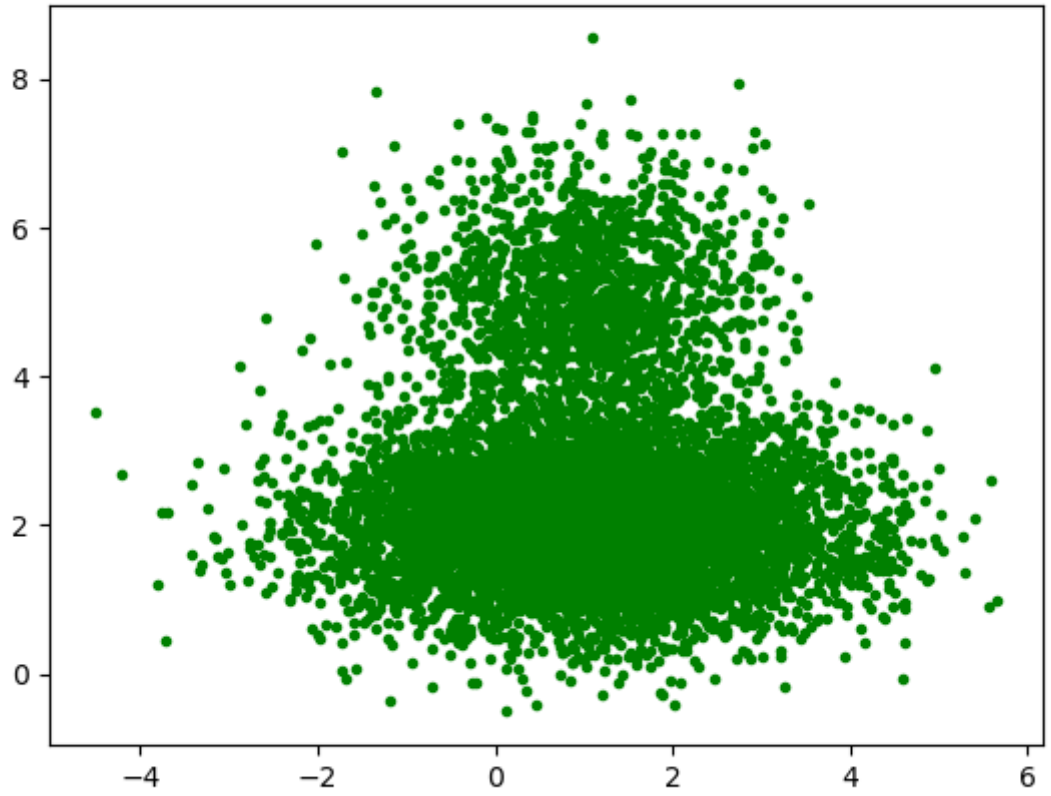
混合高斯模型拟合算法

混合模型主要是针对一个问题：某个状态ω 对应的观测数据的分布并不总是集中的。

集中的情况：



不集中的情况：



1) 首先，我们模拟不集中的数据，从两个二元正态分布中随机抽取一定比例的数据。

```
x1 = np.random.multivariate_normal([1, 2], [[2, 0], [0, 0.5]], size=3500)
x2 = np.random.multivariate_normal([1, 5], [[1, 0], [0, 1]], size=1500)
x = np.append(x1, x2, axis=0)
```

2) 初始化*K* 个二元正态分布的参数，使用数据集的信息初始化，这样也许更接近*x* 的分布状态。

3) 执行*E* 步和参数更新的操作。

4) 统计新参数下整体的对数似然，计算与前一次的对数似然相减的绝对值，若绝对值足够小则认为达到了极值点，也就是局部最优解，停止拟合。

拟合结果：

1) 在区域内选择100*100个点。

```
x1_min = np.min(x[:, 0])
x1_max = np.max(x[:, 0])
x2_min = np.min(x[:, 1])
x2_max = np.max(x[:, 1])
data_x1 = np.linspace(x1_min, x1_max, 100)
data_x2 = np.linspace(x2_min, x2_max, 100)
data = np.zeros(shape=(100 * 100, 2))
for i in range(100):
    data[i * 100:(i + 1) * 100] = data_x1[i]
    data[i * 100:(i + 1) * 100] = data_x2.reshape(100, 1)
```

2) 用拟合得到的混合高斯模型计算这些点的概率密度。

```
pre = np.zeros((100,100))
for i in range(2):
    temp = w[i] * stats.multivariate_normal.pdf(data, mean[i], sigma[i]).reshape(100, 100)
    pre = pre + temp
pre = np.rot90(pre, 1)
```

3) 利用灰度图进行可视化。

```
pre = pre * (255 / np.max(pre))
im = Image.fromarray(pre)
im = im.convert('L')
im.show()
```

根据可视化结果，这个拟合应该是有误的，但是暂时还没有分析出算法哪里有误。