Assignment 3

Part1 GMM模型与EM算法

GMM

高斯混合模型是由K个子高斯分布加权生成的概率模型，所有子分布的权重之和为1。
可以表示为

$$P(x|\theta) = \sum_{k=1}^{k=K} \alpha_k \mathcal{N}(x|\theta_k) \quad (\alpha_k \geq 0, \ \sum_{k=1}^{k=K} \alpha_k = 1)$$

其中$\mathcal{N}(x|\theta_k)$均为高斯分布，θ_k 包括 μ_k，σ_k

EM算法

使用EM算法训练GMM模型，EM算法可以分为E步与M步

- E步计算出每个样本属于第k个子高斯分布的概率

  $\gamma_{nk} = \frac{\pi_k \mathcal{N}(x|\theta_k)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x|\theta_k)}$
- M步利用E步结果，更新GMM参数

$$N_k = \sum_{n=1}^{N} \gamma_{nk}$$

$$\pi_k = \frac{N_k}{N}$$

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk} x_n$$

$$\sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk} (x_n - \mu_k)^2$$

其中，更新σ_k时，使用的μ_k是本轮更新后的

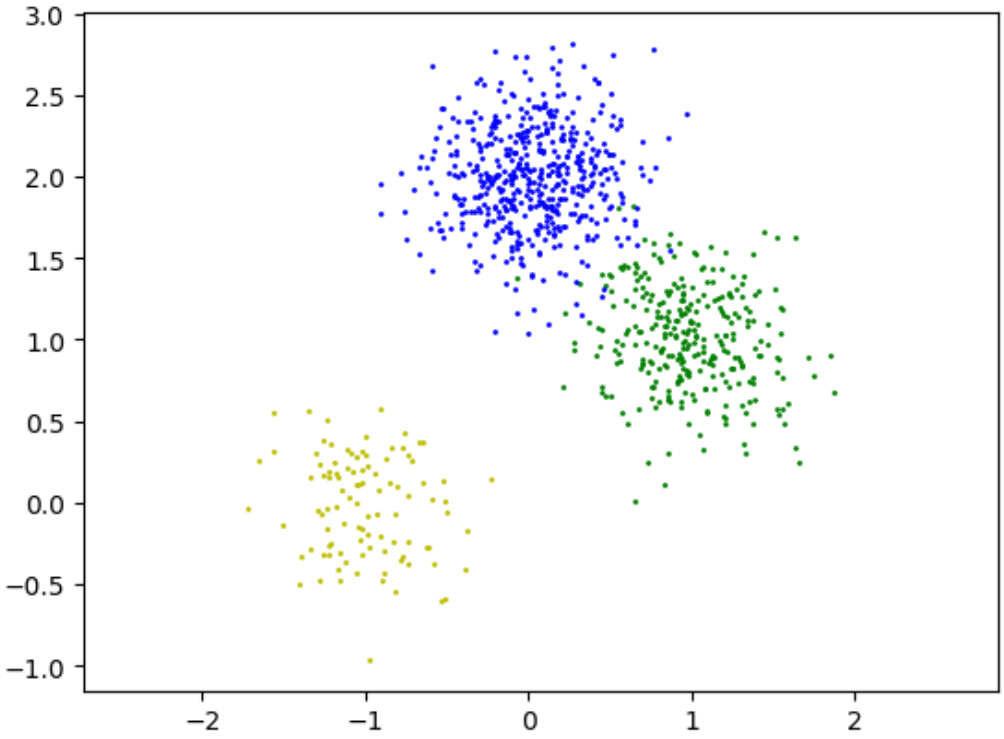
Part2 生成数据

为了后续更好的考察模型能力，选择从多个高斯分布采样获得训练集。为方便作图，使用二维高斯分布。使用的高斯分布数量、高斯分布的均值、方差、以及采样的比例均可调。

默认情况下，选用的参数如下：

```
mean = [[-1, 0], [1, 1], [0, 2]]
cov = [[[0.1, 0], [0, 0.1]], [[0.1, 0], [0, 0.1]], [[0.1, 0], [0, 0.1]]]
data_scales = [100, 300, 500]
```

使用的数据是有一定偏斜的，方差较小且都为对角阵，生成的数据如图



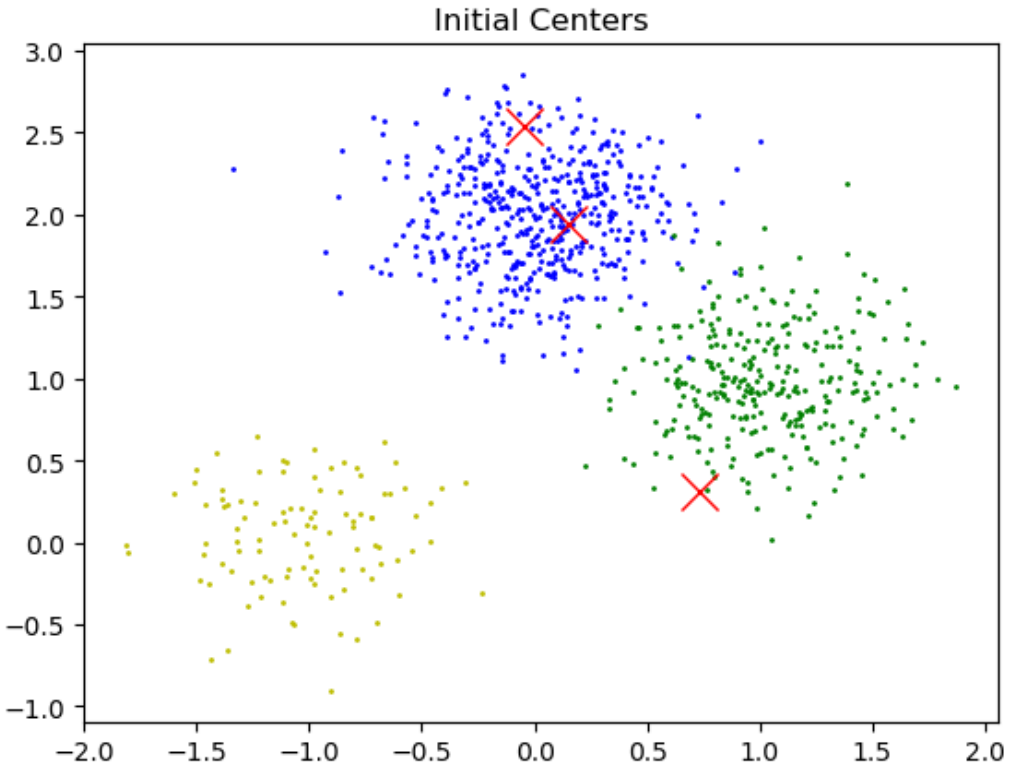
可以看到数据的边界比较清晰，仅有一小部分重合区域

Part3 搭建模型

模型初始化

- 随机初始化

  首先考虑使用随机初始化，这里的随机初始化是指从训练数据中随机取K个点作为K个子高斯分布的中心。然而，这种初始化存在一些问题。根据后续实验的经验看，当选的**K个点恰好属于K个高斯分布时**，模型能非常快的收敛，而且准确率非常高。然而随机初始化很难保证这一点，尤其是当样本不均衡时，某个分布的采样点过多时将可能所有的初始点都在这个分布中，此时模型的收敛速度和准确性都受影响。随机初始化的典型结果如下：



可以看到有两个初始点落在样本最多的分布，所有初始点都远离样本最少的分布

- K-means++ 初始化

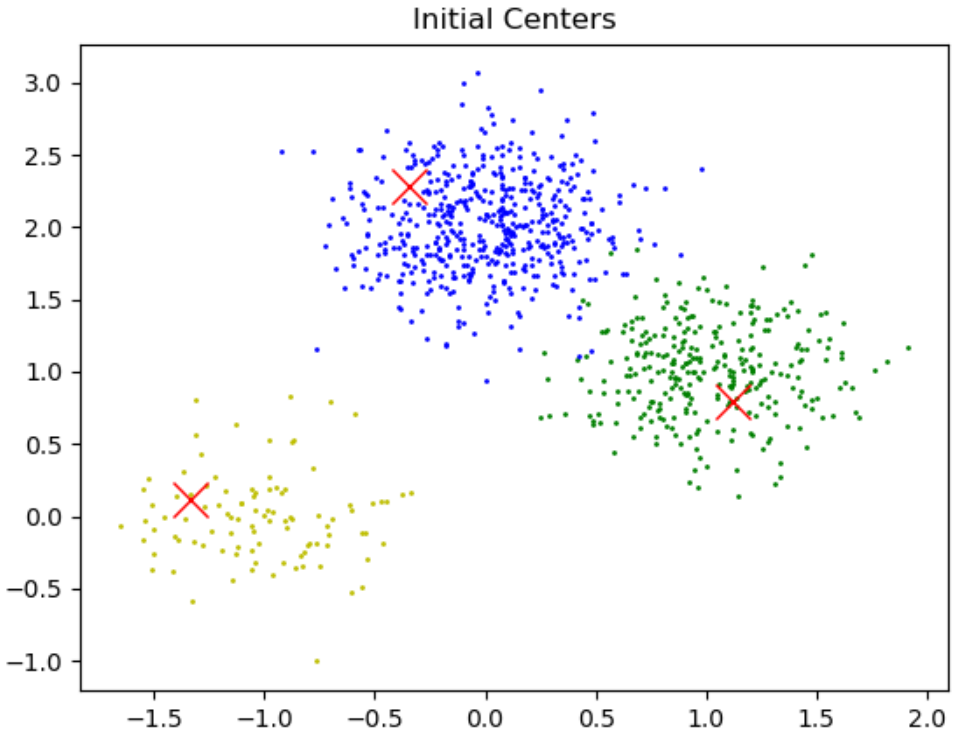
  Scikit-learn 文档中提到在初始化阶段可以使用**K-means算法**来获得比随机初始化更好的初始点，K-means 方法是一种无监督聚类方法，它的原理非常简单，就是随机选取K个点，然后将所有点按到K个点的距离归类，然后计算每个类的中心以更新K个点，直到收敛。但是，K-means算法同样需要选择初始点，因此在**样本不均衡或者运气不好**的情况下，K-means的表现会受影响，而且这是初始化阶段，需要考虑K-means运行的epoch数，如果运行过多的epoch，那么会消耗很多时间，实际聚类效果也不如EM算法；如果运行过少的epoch，在初始点选取不好的情况下得到的结果会比较差。K-means也**容易受到离群点的影响。**

  考虑使用**K-means++算法**代替K-means算法。K-means++算法是在K-means算法上的改进，算法主要有以下步骤：

  1. 从所有样本点中随机选取一个点加入初始节点集合V
  2. 计算样本中每个点到初始集合V中点的最小值，记该距离为D_x
  3. 计算每个样本被选为中心的概率 $\frac{D_{xi}^2}{\sum_{i=1}^{N} D_{ix}^2}$
  4. 按上述概率从样本中抽取一个点加入V，回到2继续添加节点直到|V|== K

  K-means++算法主要的思想就是下一个选择的点尽可能距离之前选择的点距离较远。在生成K个点后，算法结束，直接开始使用EM算法，而不需要像K-means算法一样考虑训练的epoch数与收敛条件。算法消耗的时间短，同时更不易受离群点与样本不均衡等的影响。

  采用K-means++初始化后，虽然仍不能保证K个点恰好属于K个高斯分布，但是从概率上说要远好于随机初始化。K-means++ 初始化的典型结果如下：



模型训练

按EM算法实现模型训练部分，具体见代码

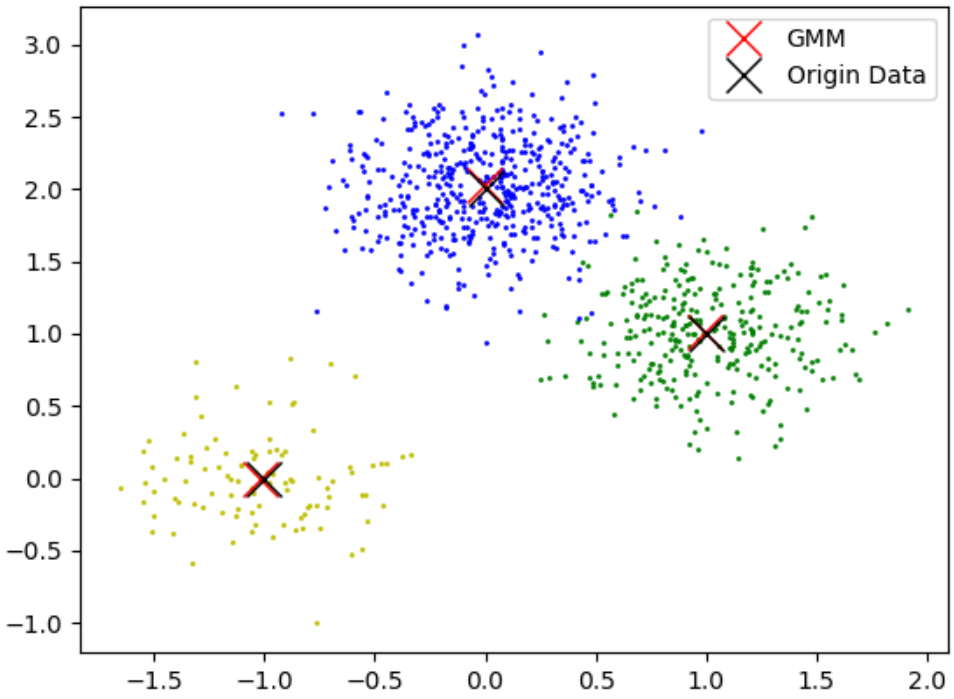
当K个点恰好属于K个高斯分布时，在默认数据集上，能在10个epoch以内收敛。当初始点选择不好时，模型收敛的epoch不固定，但大多能在100个epoch内收敛。

模型评估

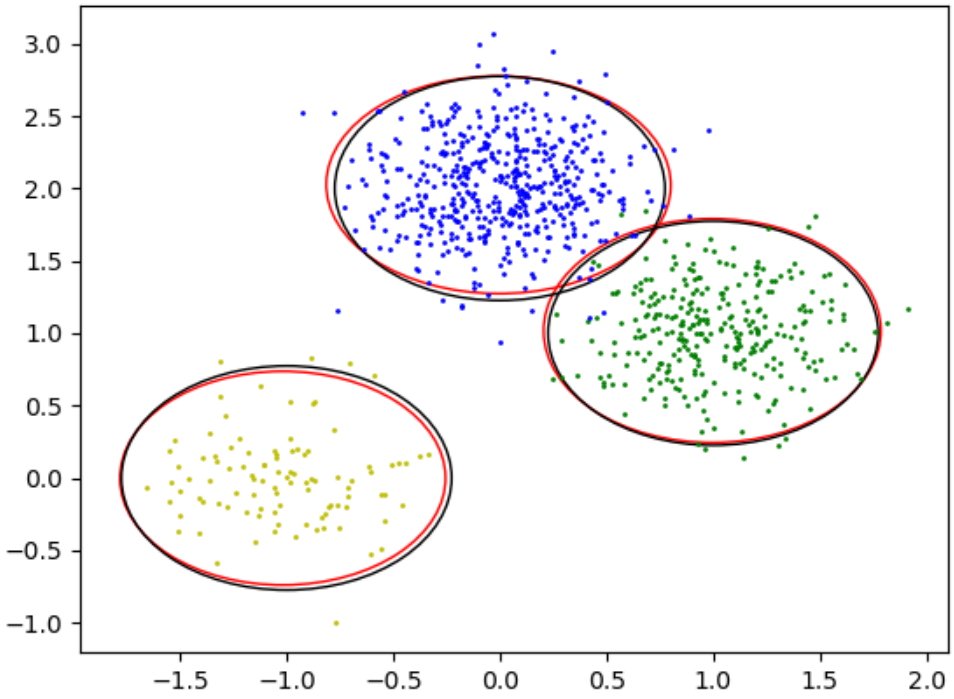
可以考虑将GMM当做**分类器**使用，对每个点GMM给出概率最高的分布作为分类结果。然后**检查分类的准确率**等指标。但是这样评估模型**存在问题**，GMM的训练过程是**无监督**的，它将样本分为K类，虽然类别数量上一致，但类别之间并**不是天然对应**，而需要人工做出判定。举个例子，如果两个分布大面积重叠，中心距离很近，那么如何判断GMM产生的两个中心点分别对应原始分布的哪一个中心点？**不同的对应方式会产生不同的模型准确率。**这就可能错误评价了模型的性能。除此以外，GMM模型的真正能力应该体现在对原始分布的拟合上而不是分类准确率上。比如K=2的GMM，对某个点能正确分类，但某一类的概率只比另一类高一点点，而在实际的分布上，这个点属于两个子分布的概率相差很大。这说明这个GMM在这一点上其实不能很好的拟合模型，但是由于**只考虑分类的准确性，这部分信息就被掩盖了。**下面展示几种我采用的评估模型的方法

- 模型训练后作出模型的参数，即模型的中心点与方差。中心点可以直接在图上作出，但方差不好直接展示。考虑使用二维高斯分布的置信椭圆来展示GMM与实际分布的方差差异。

  当初始点如K-means++部分展示的那样时，经过10个epoch，GMM与实际分布的中心如下：



GMM与实际分布的95%置信椭圆如下：



GMM与实际分布的中心几乎重合，置信椭圆也几乎完全吻合。

- 中心和置信椭圆只能作为一种展示的手段，如何**定量**的衡量GMM与实际分布的差异呢？要计算两个分布的差异，我想到了使用**KL散度**。

  KL散度的形式如下：

$$KL(P||Q) = \int p(x) \log \frac{p(x)}{q(x)}$$

  直接计算积分是困难的，并且对于GMM可能没有闭式解。转化为求期望并使用蒙特卡洛方法

$$KL(P||Q) = E_x \log \frac{p(x)}{q(x)} = \frac{1}{N}\sum_{i=1}^{N} \frac{p(x_i)}{q(x_i)}$$

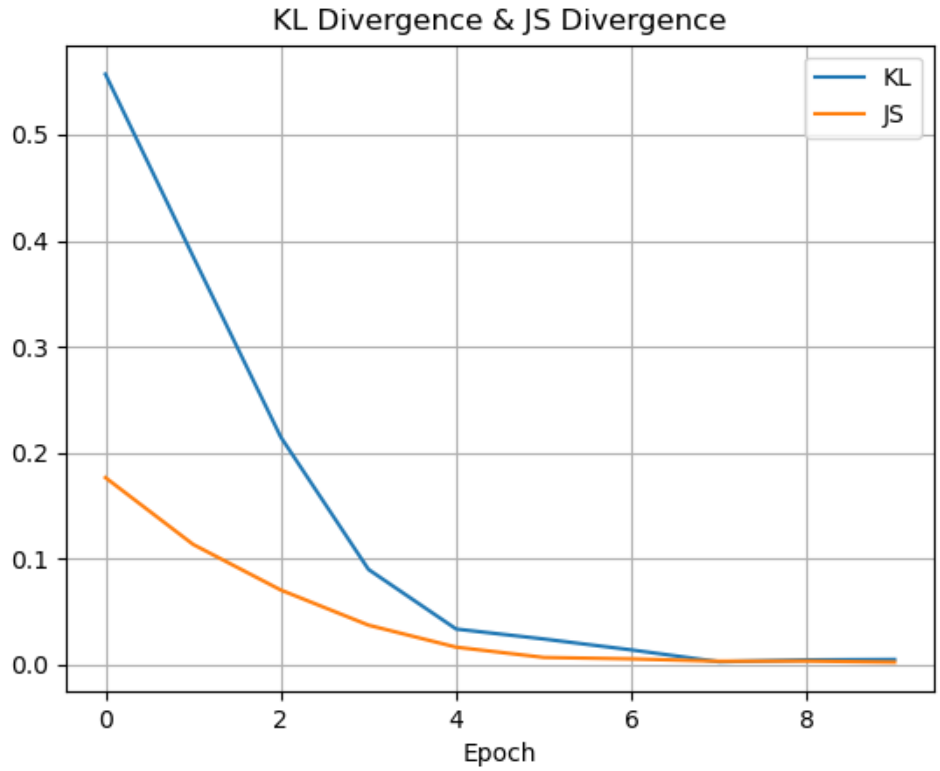
每一轮epoch训练后，计算GMM与实际分布的KL散度，即可定量显示模型表现的变化与接近程度。

- KL散度是不对称的，而两个分布的重合程度这件事本身应该是对称的。一种思路是将再求KL(Q||P)。现在采用一种类似的策略，使用**JS散度**，JS散度综合了KL(P||Q)与KL(Q||P)，具体形式如下：

$$JS(P||Q) = \frac{1}{2}KL(P||\frac{P+Q}{2}) + \frac{1}{2}KL(Q||\frac{P+Q}{2})$$

JS散度是对称的，取值在0-1之间。并没有考虑使用Wasserstein距离，因为不需要考虑完全无重叠时的梯度消失问题。

仍然在上文初始点条件下，10个epoch的KL散度和JS散度变化如图：



KL和JS散度的变化趋势一致，均快速下降，最终在10个epoch的时候均收敛到10^{-3}以下，结合置信椭圆，KL，JS散度的值比较好的体现出GMM和真实分布的重合情况。

Part4 讨论

不同初始化方式的差异

这里定量讨论K-means初始化方法与随机化初始化方法相比是否确实更优，在相同数据集（默认数据集）下，分别采用随机初始化与K-means初始化，测试模型收敛所需的epoch数

经过观察发现，当不同模型训练收敛时，JS散度会在0.01以下浮动．以0.01作为阈值，如果连续 5 个epoch中JS散度均小于0.01则认为模型已经收敛

对每种初始化方法重复运行10次取平均值，得到的结果如下:

初始化方法	收敛平均epoch数
随机初始化	26.2
K-means++初始化	16.7

可以看出，K-means++初始化的表现要明显优于随机初始化，而且最坏情况K-means++也优于随机初始化．随机初始化中出现了2次需要50个epoch才能收敛的情况，K-means++初始化则只出现了一次需要30个epoch收敛的情况

不同规模样本的差异

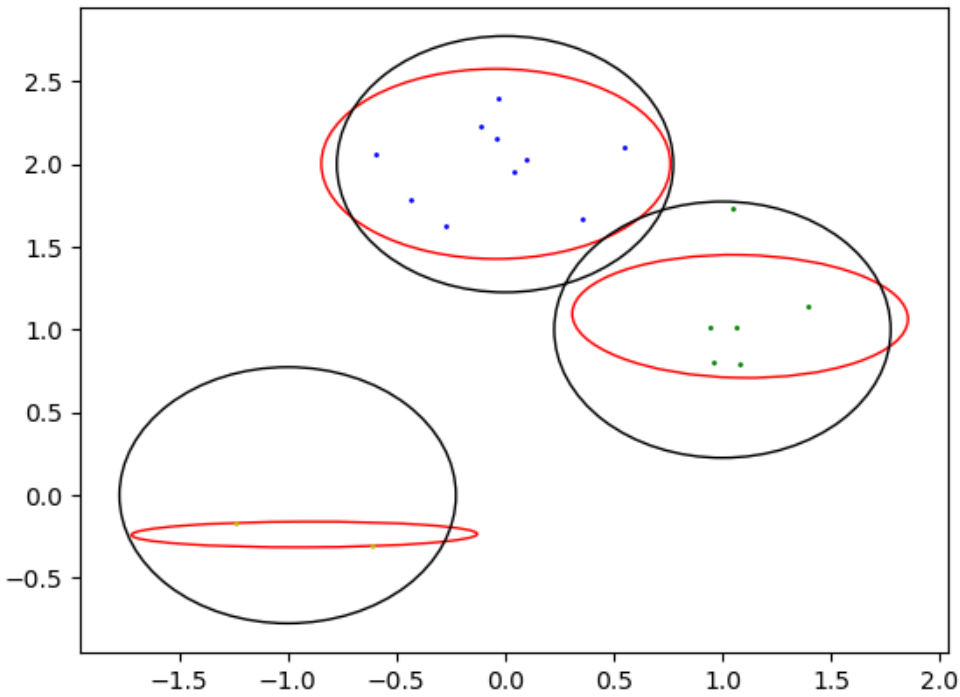
构造了总样本数分别为10，20，50，100，200，500，1000的样本，样本集的划分均与默认数据集一致，用来生成样本的子高斯分布与默认数据集也一致

测试模型收敛所需的epoch数，仍以0.01作为阈值，如果连续 5 个epoch中JS散度均小于0.01则认为模型已经收敛

对每种初始化方法重复运行10次取平均值

在测试中有以下几个发现：

- 样本量为10和20时，对每次训练，JS散度先短暂下降,然后很快收敛于0.5，这表示测试模型无法与原始分布很好的拟合．观察置信椭圆和样本中心后发现，即使在样本量为10，单个类别仅有一个点的情况下，模型也能基本正确的拟合中心．但是，特别对于点数少的类别，正确估计出方差几乎不可能，因此JS散度较大．典型的样本中心与置信椭圆图如下



- 当样本量为50以上时，中心与置信椭圆都能较好拟合
- 讨论模型的收敛速度意义不大，因为大部分模型都能在10-20个epoch之间收敛，与样本规模关系不大，样本规模主要对训练时间有影响
- 样本规模对最终收敛时拟合的效果有影响，主要影响方差的估计，样本量越大，收敛时的JS散度越小，模型拟合程度越好

Part5 用法

```
python source.py
```

默认数据集与子高斯分布参数，默认20次epoch

运行时会作出初始点选择图（initial points），中心拟合图，95%置信椭圆图，以及KL，JS曲线

运行时会输出GMM训练后的参数以及最终的KL，JS散度值