Assignment1 Report

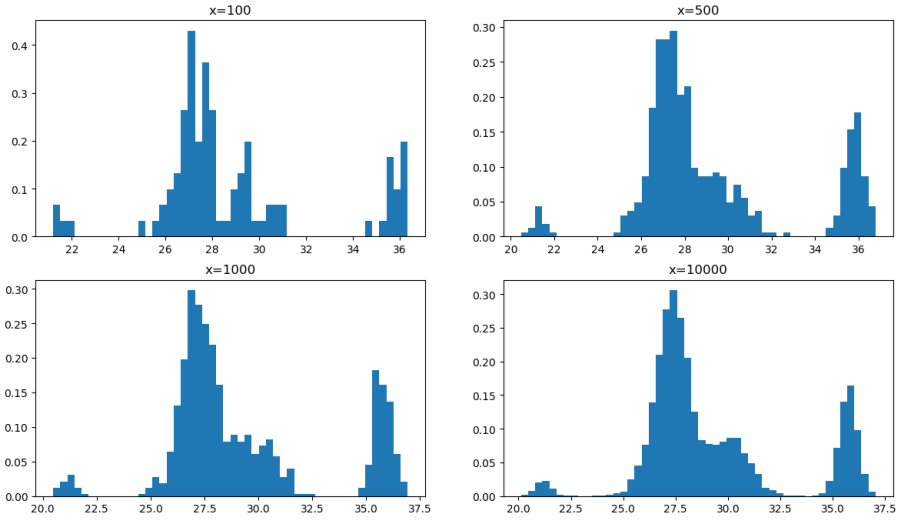
18302010047 戴予淳

2019 年 3 月 19 日

1 Part 1

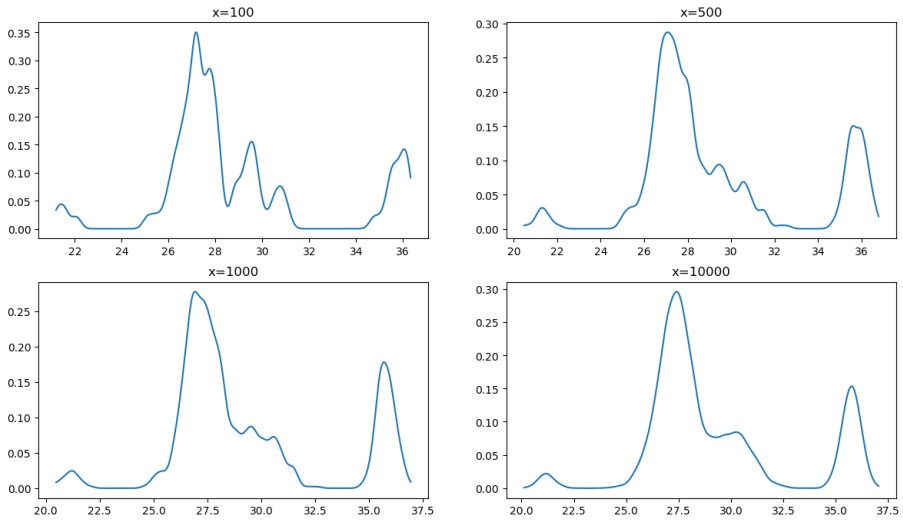
为了研究数据规模对算法的影响，可以很直观的把对应的图都画出来进行观察。

1、直方图（Histogram）：



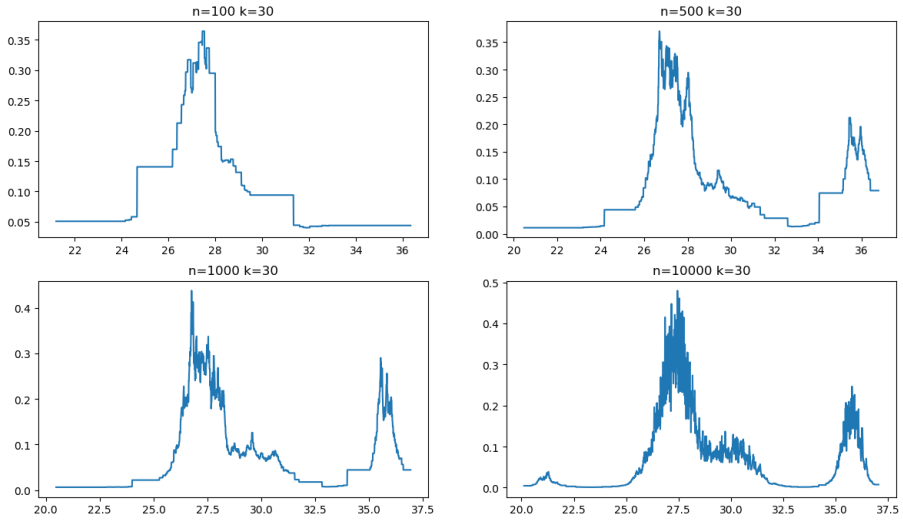
此图即为生成的一维数据出现的概率分布图，可以发现，随着数据规模增大，整个形状越来越平滑，也能更好的模拟生成数据程序中提供的概率分布图。

2、核函数估计（Kernel Density）



此处是用高斯函数为核函数进行概率密度图的绘制，可以发现，随着数据规模的增大，图像中波动减少，更加的平缓，形状也更接近生成数据的概率分布图。

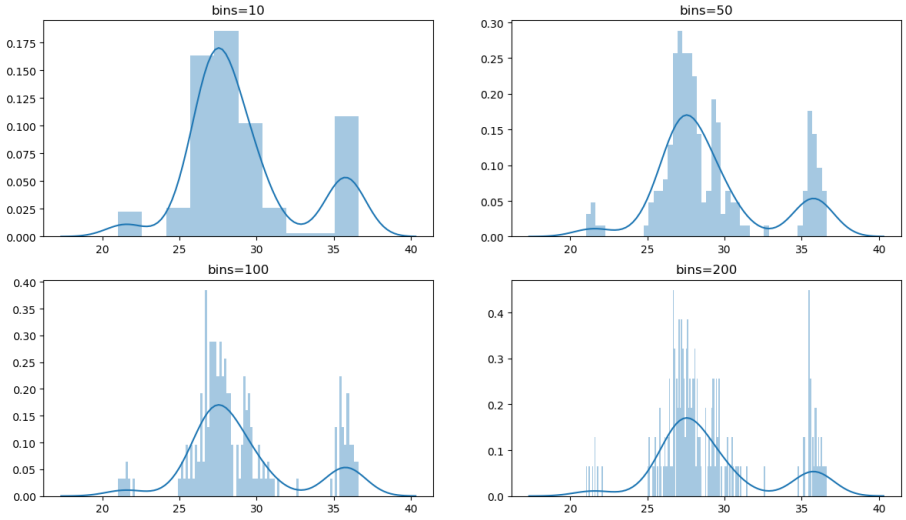
3、K近邻估计（K-nearest neighbors）



此处使用K近邻的方法来估计概率密度，可以发现，当数据规模增大而K保持不变时，图像有变得更加尖锐波动的趋势，异常的高峰越来越多。

2 Part 2

要研究直方图中桶数取多少时比较合适，可以考虑首先画出直方图，再画出关于这个直方图的拟合曲线，再将该曲线与原数据的曲线相对比，误差越小就证明桶数选择越合适。我利用的是seaborn库中已经封装好的函数来对直方图进行曲线拟合，以下是几组不同的桶数的前提下的拟合情况：

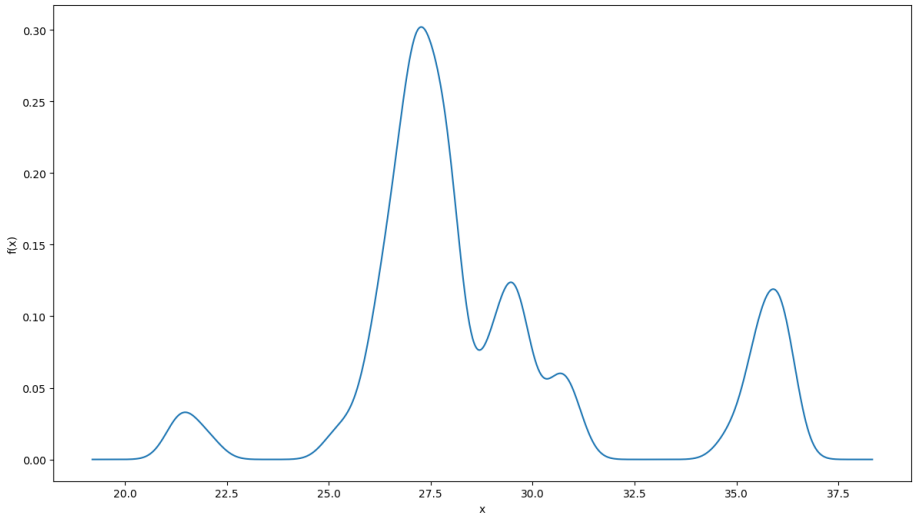


可以发现，当桶数比较小的时候，不太能达到概率密度估计的效果，只能呈现主体的峰，一些比较小的峰就被去掉了；而当桶数比较大的时候，直方图波动非常大，导致拟合曲线的峰大大降低，也无法达到估计效果。因此桶数应该选择一个适中的值。

根据我的估计来看，我认为bins=20 ~ 30是比较理想的值。

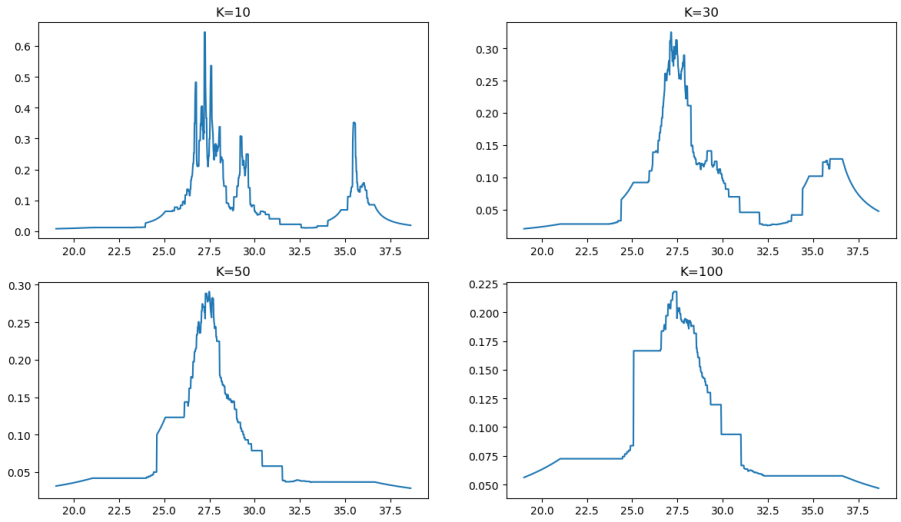
3 Part 3

之前研究数据规模对Kernel Density算法的影响时，出于方便考虑，使用了一个已经封装好的库sklearn，它可以帮你很方便的求高斯函数这类的。而在part3中，我手动实现了一下这个算法，其实也不难，利用numpy中的linspace函数将给定数据的值域划分成1000段，然后对于每个点计算高斯函数，用这些点来拟合概率密度曲线就好了。关于h的取值的话，当h比较小的时候，曲线会产生剧烈波动，当h比较大时，曲线会变得平滑。可能与我的算法不够精细有关，我模拟出的图像中主峰右端永远有两个峰而不是像造数据的程序中那样只有一个峰，而且最右侧的峰的高度不是很高，调参好像没法解决这个问题...经过尝试后我觉得**h=0.35** 时得到的结果是比较好的，效果如下：



4 Part 4

之前同样使用的是sklearn库来探究数据规模对KNN算法的影响，在这里重新实现了一下，首先将值域划分为1000份，然后对于每一个点，暴力求出每个样本到其距离的绝对值，然后排序取前K个然后估算概率，但是需要注意的是这是在数据范围为200时的可行方法，当数据范围比较大，如1w时，应该要借助KD-tree来进行K近邻查询，那个比较复杂就没有实现了。以下是我在几个K取不同值的情况下得到的概率曲线：



关于K近邻方法得到的概率和，我在程序中粗略计算了一下，在上面四个图中加得的概率和分别为：**1.5009635337834348，1.6011280199543205，1.4395224467987946，1.8082501631131365**可以竟然已经超过1了...当然这与我计算比较粗略有关（没有计算大范围$(-\infty, \infty)$上的积分，区域划分也不够细，除法时也有一定的误差）

关于K近邻算法对概率积分无法得到逼近1的概率，我想是因为我们利用$p(x) = \frac{k/n}{V}$来代表某处的概率，其中V代表一个体积微元，这个算法中就是认为在这个体积微元中概率密度是不变的，但是实际上它可能是会产生变化的，这么估计并不精确，事实上，这样估计概率密度只有在样本量特别大（即n特别大，那时候整个的分布就比较稳定，随机性比较小）时是比较合理的。在这里$n = 200$样本量不够大，这样估计就会产生问题，积分出来就非常不精确。