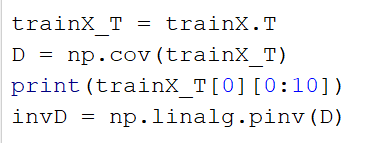
1. 使用马氏距离

（1）计算协方差矩阵的伪逆



（2）在kd是否需要跑另一个分支的时候，同样使用相应的马氏距离来判断

（3）在本实验的代码中，需要在初始化kd树的时候传入参数字符串以及对应的矩阵



如果普通距离，则为simple：

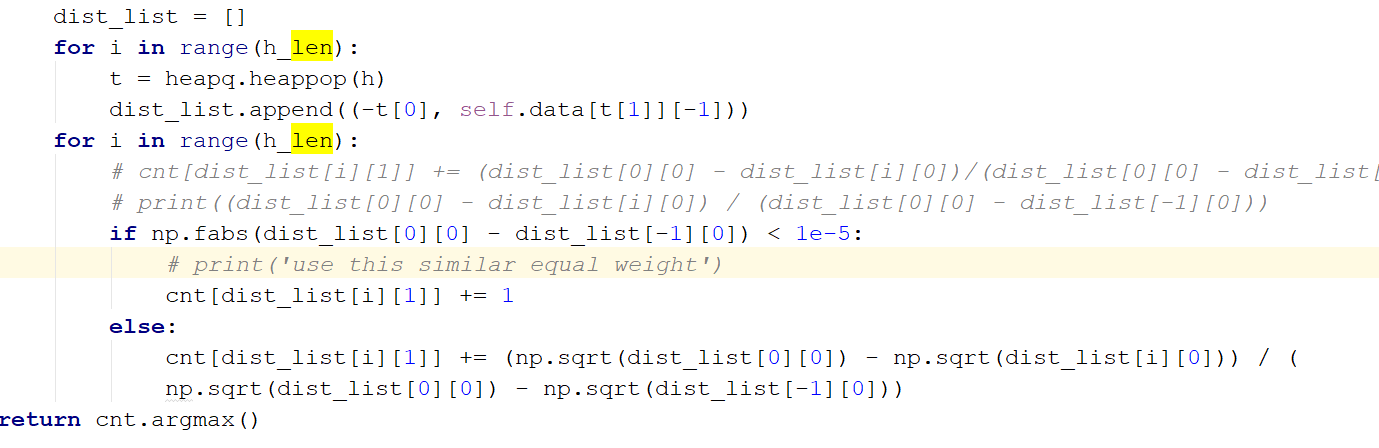


2.使用wknn

（1）在统计最近的k个点的时候，每个点的权重和它的距离相关，公式为：



（2）在代码的实现中，是一样的：



需要注意的是，如果所有的k个点的距离恰好一样，那么是特殊的情况，分母会为0，此时只需要所有的权重都是1就可以。并且在我们的1024个维度都是0或者1的情况下，很容易出现距离都相等的可能性。

（3）在本实验的代码中，需要在初始化kd树的时候传入字符串：



不是wknn的时候，不传way的值即可。

3.实现了heap的kd树，优化策略为简单的半径和轴长的比较. 跑不动，引入pca

（1）使用kd树来进行优化，每次分成两堆，在选择维度的时候，简单的使用+1%维度的方法，在查找的时候，使用简单的比较坐标轴距离和目前k个中最大距离的策略。K个最小距离用heap来维护。

（2）使用kd树优化，如果使用原始数据，1024维，并且每维度都是0或者1，那么速度会比线性的慢的多，因为每个维度都是0或者1使得kd树的优化策略几乎没有用，而使用了heap和递归也会使得当kd树退化到线性的时候没有直接线性快。因此为了体现kd树的价值，我对1024维度的引入pca降维度到32

4.5种kd树的比较，以及选取hyperparameter k 的策略

A。使用pca的数据

一共5个knn：

①Sklearn中的knn，作为一个比较好的knn的对比；

②使用欧式距离，不用wknn的正常knn，基于kdTree实现；

③使用马氏距离，不用wknn的正常knn，基于kdTree实现；

④使用欧式距离，使用wknn的knn，基于kdTree实现；

⑤使用马氏距离，使用wknn的knn，基于kdTree实现；

（1）knn的k的选取：

采取把train集分成80%的train和20%validation集，找到最好的k，然后再用这个k训练100%的train，最终5个knn分别的最好的k值为[3 3 1 9 5]

（2）在test集合上的测试：

正确率的定位为：预测标签和实际标签一样的/总个数

正确率分别为：

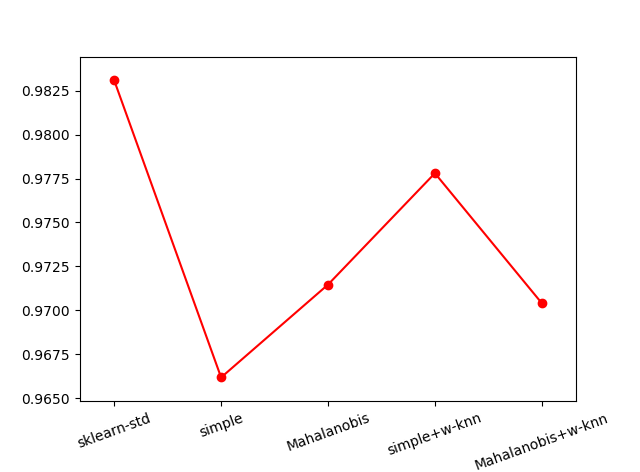
0.9830866807610994

0.9661733615221987

0.9714587737843552

0.9778012684989429

0.9704016913319239



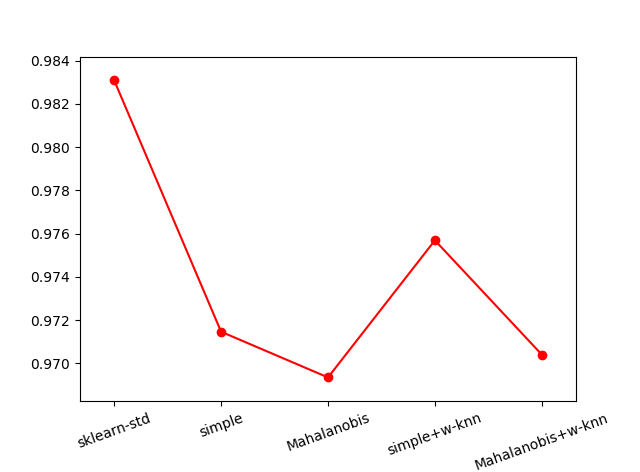
图一 使用pca降到32维，使用[3 3 1 9 5]分别为k值

（3）结果分析

A。Sklearn的效果最好，在没有传入权重的参数下，sklearn的方法应该和simple的是一样的，但实际上sklearn的更好，应该是sklearn做的有些优化，对于经过pca变换之后的这样的数据处理的更好

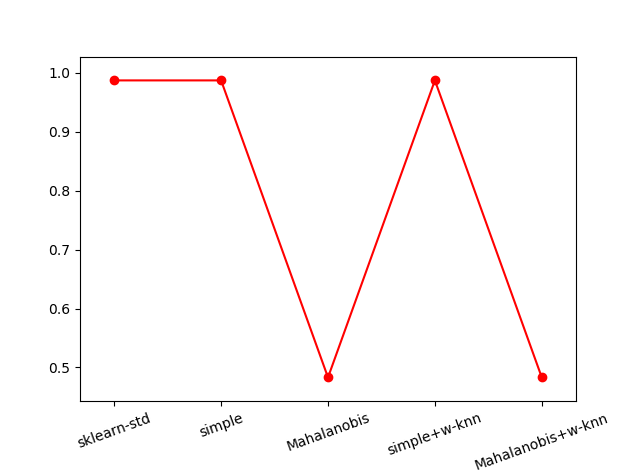
B。在使用马氏距离之后，wknn的效果变差，而不正常的knn的有提高，说明马氏距离可以更好的反应距离之间的大小关系，但是因为做了伸缩变换，使得距离之间的比值关系并没有变换前的比例关系更加的能够反映出“属于某一类的概率”，因此使用了wknn反倒变差了。

C. 实现的最好的情况是欧氏距离+wknn，并且结合选取了k均为5的下面的结果，发现的确马氏距离的使用不稳定，并且依赖于knn的结果选取策略，而对于欧氏距离，wknn的策略能够稳定的提高，因此wknn和二阶矩之间的叠加是不相互矛盾的。



图二 使用pca降到32维，使用[5 5 5 5 5]分别为k值

B。使用原始维度（1024）的结果分析



图三 1024维，使用[3 3 1 9 1]分别为k值

0.9873150105708245

0.9873150105708245

0.48308668076109934

0.9873150105708245

0.48308668076109934

结果非常特殊：

（1）sklearn的knn和实现的欧式距离，不用wknn的knn结果一样，因此对于这样特殊的数据，sklearn没有做任何决策上的优化。

（2）使用马氏距离会变差，也和维度上取0和1二值可能有关

（3）使用wknn没有用，因为在实现中也发现，这种情况下的k个最小维度距离很多都是相等的，因此权重会被特判为等价的，所以wknn没有用。

（4）使用原始数据虽然最高的正确率会高，但是在时间上不理想，以及因为数据的二值特殊性，使得实验探讨的马氏距离和wknn的效果不理想，因此不作为主要实验结果分析。