Training：

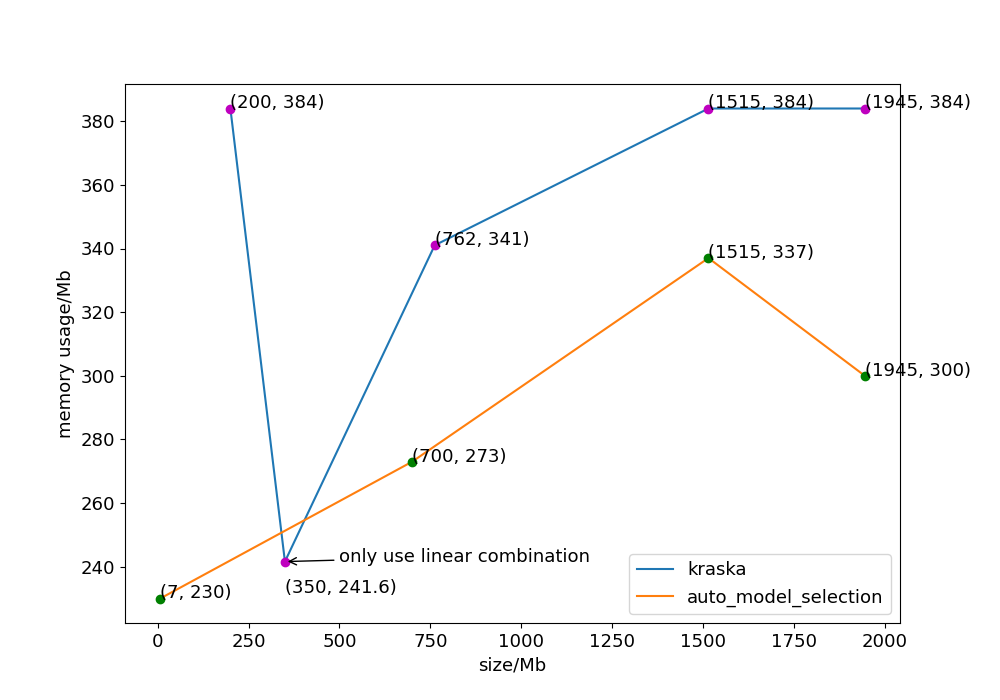
我们将auto-model-selection与kraska的rmi模型进行了对比试验。在rmi模型中，读入的数据被指定是二进制文件，可以直接读取数据进行训练。而在我们的模型中，数据集是csv格式，需要首先经过数据处理，加入噪声，并分割为训练集和测试集，再进入模型训练。对于大规模数据（200M条以上），这一过程比较耗时，内存消耗严重并且对于学习索引的贡献不大，我们认为这是一个在未来工作中值得优化的方向。在此阶段，rmi进行各种尝试，组合出性能最好的多层模型；我们则是训练多个回归模型，挑选出性能最好的模型用于今后的查询工作。

Result：

我们找到了mit与kraska合作的这一开源项目所提供的数据集。这部分数据集是有序的。其中包括一个大小为700Mb的，有200M条图书信息的32位unsigned int整数数据集和一个大小为1.48Gb，含有200M条wiki信息的64位unsigned int整数数据集。为了测试不同大小的数据集下模型的表现，我们还生成了200M的随机分布数据集，350M的指数分布数据集，以及1.9G的正态分布的数据集。

实验一：内存占用对比

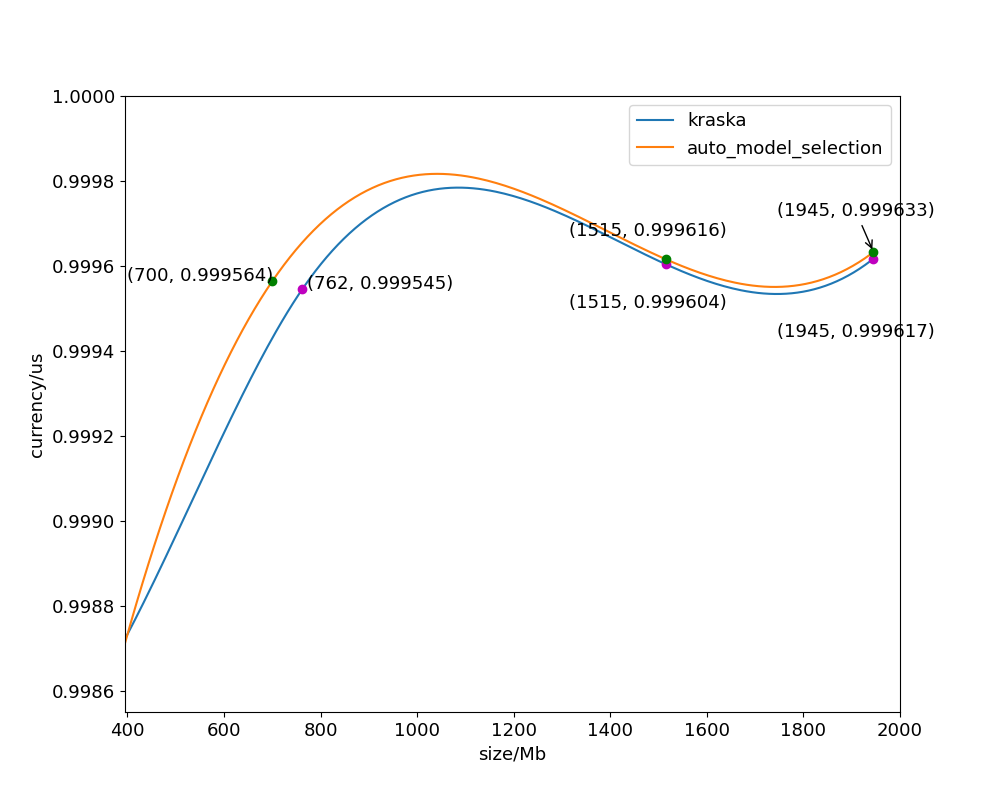
首先，针对不同大小的数据集，我们对模型的内存占用情况进行对比试验。Rmi中可供选择的模型都是单个内存占用低，相对简单的弱学习器回归模型或者多阶样条函数，采用多层叠加的方式提高总体效果。在我们的auto\_model\_selection模型中，我们采用占用单个内存占用相对较高，但是拟合效果很好的多种回归模型。实验效果如下图所示：



图中，横轴是选择的数据集的大小，单位为Mb，纵轴是内存占用情况，单位为Mb。橙色线条代表我们的自动选择模型，蓝色的是kraska的rmi模型。图中的点是我们记录的实验结果在实验中，对于随机分布（200，384），wiki（1515，384），正态分布（1945，384），三者采用优化等级最高的rmi模型与我们训练得到的最优模型进行了对比试验。实验表明，rmi优化等级最高情况下，内存占用基本固定为384M，为4层rmi（4-stage rmi）。此时由于rmi的层数较多，大量模型的堆积使得其内存占用情况超过了我们的自动选择模型挑选出的单个回归模型的内存占用。因此我们的模型在此方面表现得更加优秀。而另一方面，为了权衡模型其他效果（准确率和查询时间）与模型内存占用哪一个更需要优化，我们设计：将我们挑选出的模型与rmi中只用内存占用最低的线性回归模型叠加而成的组合层次结构进行对比试验。观察到点（350，241.6）处，rmi的内存占用略低于我们的自动选择模型。然而在接下来的实验中，我们将会观察到，这点处的查询时间rmi的时间约等于我们模型的二倍！（见实验三）所以我们在内存上的优化的优越性得到了保证。

实验二：查询准确率对比

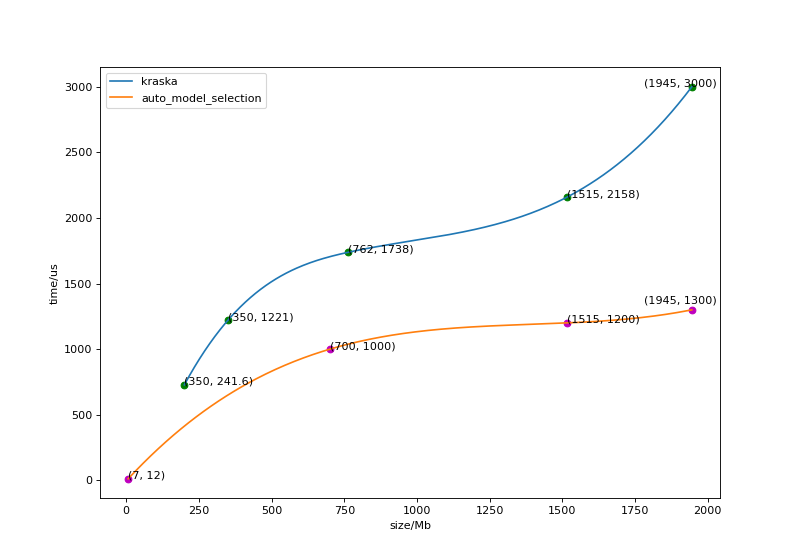
下一步，我们对于查询结果准确率进行了对比试验，实验结果如下图所示：



可以观测到在400M到1200M之间，我们的模型准确度比kraska模型都有着较大程度的提高，尤其是在600M到1000M之间，能高出约为0.0001。在1200M之后，自动选择模型的准确率略微高于rmi模型。这是由于大量数据集训练下，我们采用的优秀的回归模型比如梯度提升树（GBDT）与极限树（ExtraTree）等，能有更丰富的训练数据集进行学习，从而获得更好的训练效果。同时，数据集增大后，自动选择模型中的数据经过重分布后，附加在key上的模型标签更加多种并且可靠，这些数据输入到分类器后，分类器训练效果得到了提高，因此在查询时，总体的准确度得到了一定程度的提高。总体来看，我们的模型的准确率在全程都优于rmi。

实验三：查询时长对比：

最后，我们对二者的查用用时进行了对比实验。由于rmi模型经过4层的学习，查询的结果是逐层输出的。而我们的自动模型选择模型，查询结果不用经过所有回归模型的训练，直接到分类器中寻找最优匹配的回归模型然后进行训练，并且模型的性能更高，因此在理论上，我们的查询时间更短。为了验证这个理论，我们对比试验的结果如下图所示：



正如实验一最后讲述的，在（350，241.6）处，对于350M的数据集，rmi最终查询时间约为我们模型的二倍。根据上图可以看出，rmi的模型在1500M之前约为对数增长趋势，在1500M之后斜率增大，有近似于指数的增长趋势；对于我们的模型，保持着略慢于对数增长的趋势，这是由于我们某些优秀的回归模型，其本质是树形结构。观察到，在1500M之后，点（1515，2158），（1945，3000），这两处的查询时间约为同等大小数据集下，我们模型用时的两到三倍，因此，面对容量较大的数据集，我们的模型在查询时间上有着巨大的优势。