Pipeline的steps参数设定数据处理流程。格式为('key', value)。key为该step设定的名称，value是对应的处理类，最后通过list将其传入。前n-1个step中的类必须有transform函数，最后一个可有可无，且Pipeline继承了最后一个类的所有方法。

|  |
| --- |
| 1. **from** sklearn.pipeline **import** Pipeline 2. **from** sklearn.pipeline **import** make\_pipeline 3. **from** sklearn.linear\_model **import** LinearRegression 4. **from** sklearn.decomposition **import** PCA 5. **from** sklearn.datasets **import** load\_iris 7. iris = load\_iris() 8. pipe = Pipeline([('pca',PCA()), ('linear',LinearRegression())]) 9. pipe.fit(iris.data, iris.target) |

make\_pipeline是Pipeline的简单实现只需传入每个step的类实例即可。

FeatureUnion对特征进行并行化处理，每一个step分开计算，最后将计算得到的结果合并到一块并返回一个数组。

|  |
| --- |
| 1. **from** sklearn.pipeline **import** FeatureUnion 2. **from** sklearn.preprocessing **import** StandardScaler 3. **from** sklearn.preprocessing **import** FunctionTransformer 4. **from** numpy **import** log1p 6. iris = load\_iris() 7. steps = FeatureUnion(transformer\_list=[('standard', StandardScaler()), ('ToLog', FunctionTransformer(log1p))]) 8. data = steps.fit\_transform(iris.data) |

0817

通过模型堆叠，能基于泛化性能较弱的学习器构建出较强的学习器。

除了对简单的线性模型进行集成学习，可尝试使用xgboost或lightgbm这些提升算法，其本身就带有模型叠加的思想。若成绩还是不够理想，由于数据特征的质量决定了模型拟合的上限，所以可继续对原数据运用特征工程构建新的特征，然后再用集成算法对结果进行预测。

集成学习中，之前只是对最大投票法、平均法、加权平均法比较熟悉，而对Stacking和Blending认识不足，通过这次的视频讲解，以后可以更多的尝试这两种集成学习算法。此外，还得补补mlens这个库，能轻松实现集成学习，感觉就很爽。

0818

这周主要学习了特征工程和模型集成学习方法。特征工程除了常规的特征预处理之外，更加了解了特征选择的一些方法，如Filter方法、Wrapper方法和Embedded方法。再配合老师大量结合sklearn给的案例，不懂或者忘记了的时候能够时时去回顾。

学习了特征工程之后，这周又学习了模型集成学习方法，简单的如最大投票法、平均法和加权平均法，复杂的如Stacking和Blending。

答案问题：在使用算法的时候，希望老师能给一些对应算法的讲的比较好的博客文章推荐推荐，给个链接，让我们用的时候不仅仅只是调用函数库而已。

0820

sale\_train\_v2可以通过shop\_id，item\_id，item\_category\_id来合并items.csv、item\_categories.csv、items.csv和shops.csv。预测目标item\_cnt\_month – 商品月销量，需要根据item\_cnt\_day按月统计所有商店里每件商品的销量。为了简化模型，官方将item\_cnt\_month的取值范围限制在[0, 20]。

销量跟售价有着密切的关系，如售价越高的商品销量越低，反之亦然。可先对sale\_train\_v2的数据分析销量和售价的关系。

所有商品按月份统计销量，观察销量的走势，可能存在周期性规律。

仔细观察shops.csv数据，shop\_name特征通过空格进行分隔，第一个字段是城市名，第二个字段（如果有的话）是店铺的类型，最后的字段则是店名。然后对抽取出来的城市名、店铺类型进行one-hoe编码。

items\_categories.csv的item\_category\_name字段按’-‘进行分隔，获取商品的主类型和子类型，再进行one-hot编码。

参考链接：<https://www.jianshu.com/p/1f6eef8a86fd>

0823

考虑到销量和售价有着密切的关系，将商品的价格也加入特征列。但是，由于test数据中有较多的shop\_id和item\_id在train数据中没有出现过，导致按baseline的操作方式处理数据时，有很多数据需要进行填充，加了商品价格特征之后，预测的结果反而变差了。

0825

与第一次Kaggle的Predict Hourse Price相比，这次的比赛所分析的数据给的文档比较多，各文档之间通过shop\_id、item\_id、item\_category\_id联系在一起。此外，就train数据和test数据而言，其原始数据包含的特征比较少，而且数据几乎没有缺失值。我们需要结合其他的几个文件来构建具体预测所需要的特征。由于该比赛的预测目标是一个月的销量，即item\_cnt\_month，但是提供的数据都是item\_cnt\_day的，所以需要再构建需要预测的变量，其不像Predict House Price的预测变量那么直观。

最后，最重要的是首次接触了涉及时间序列预测的问题。按月份构建每个id对应的每个month的销量，然后将预测月份的前一个月作为训练数据集的预测目标，用来训练模型，最后再用完整的历史销量数据作为特征放入模型对未来的销量做出预测。

总的来说，这周新的内容就是涉及到了时间序列问题的处理方式。