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。