***说明：***

***由于算法需要所有值全为数字类型（除了分类算法的分类标签），所以数据预处理需要提供可以把非数字类型转化为数字类型的预处理方法（one-hot，label-encoder等等）***

***一共有三种算法：***

***线性回归***

***SVM分类***

***决策树分类***

***接口说明：***

***全为post接口***

***调用每种算法，都需要传输两个文件，train和test，以及一系列模型的参数，然后等待训练完成之后，需要调用另外两个接口获取model.model以及encoder.model***

***具体的接口细节***

1. ***/linearRegression***

***train: 训练使用的文件，csv***

***test: 测试使用的文件，csv***

***loss: 损失函数 str 可选的值有***{***’squared\_loss’ ,‘huber’, ‘epsilon\_insensitive’, or ‘squared\_epsilon\_insensitive’***} *默认传*** *’squared\_loss’***

****max\_iter: *最大迭代次数 int 默认传1000*****

*****shuffle: 是否打乱数据 bool 默认传True*****

****tol: 停止的误差标准** *float, 默认传1e-3***

*****返回预测的百分比误差*****

1. *****/linearRegressionPredict*****

*****训练完成后传入模型文件和希望预测的数据文件可以使用模型预测*****

*****model 模型文件*****

*****predict 希望预测的数据文件，csv*****

*****返回预测后的结果文件*****

1. ***/decisionTreeClassfier***

***train: 训练使用的文件，csv***

***test: 测试使用的文件，csv***

*****criterion: 决策树分裂选择的评估指标 str 可选的值****{“gini”, “entropy”, “log\_loss”}, 默认传”gini”***

*****max\_depth:*** *最大深度 int, 默认传1000***

*****max\_leaf\_nodes:*** *最大叶节点数量**int, 默认传1000***

***返回预测精度***

1. ***/DecisionTreeClassifierPredict***

*****训练完成后传入模型文件和希望预测的数据文件可以使用模型预测*****

*****model 模型文件*****

*****encoder 标签编码文件*****

*****predict 希望预测的数据文件，csv*****

*****返回预测后的结果文件*****

1. ***/SVMClassifier***

***train: 训练使用的文件，csv***

***test: 测试使用的的文件，csv***

*****C: 正则化参数*** *float, default=1.0***

*****kernel: 核函数 可选值****{‘linear’, ‘poly’, ‘rbf’, ‘sigmoid’, ‘precomputed’}, default=’rbf’***

*****tol: 停止的误差标准*** *float, 默认传1e-3***

*****max\_iter: 最大迭代次数 int 默认传-1（表示无限制*****

*****返回预测精度*****

1. ***/SVMClassifierPredict***

*****训练完成后传入模型文件和希望预测的数据文件可以使用模型预测*****

*****model 模型文件*****

*****encoder 标签编码文件*****

*****predict 希望预测的数据文件，csv*****

*****返回预测后的结果文件*****

1. ***/getModel***

***返回模型文件 model.model***

1. ***/getEncoder***

***只有决策树和svm分类模型需要下载labelEncoder***

***返回labelEncoder encoder.model***

预处理部分

1. <http://127.0.0.1:12000/intro>

该接口返回一个各种算法的描述，这个描述文件以JSON格式存储在后端的intro.json中。格式参考赵先生要求的格式编写。

需要注意的是，相关的描述图片和GIF需要前端自己拷贝到目标文件夹直接引用，而不是向后端索取。JSON中提供对应图片的文件名。所有资源都在后端文件夹的/demos中。

1. <http://127.0.0.1:12000/upload>

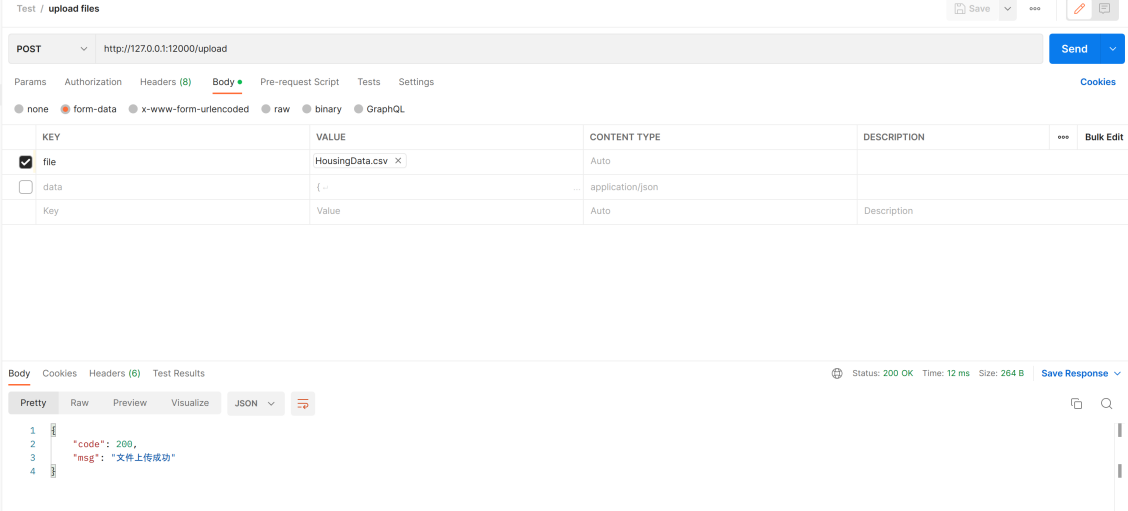
该接口用于上传文件，上传的文件会被存储为/src/data.csv,仅支持csv文件格式的上传。

使用form表单传输文件：

{

“file”: [file\_to\_be\_selected]

}



如果文件读取失败，则会返回：

{

'code': 400,

'msg': "文件读取失败"

}

1. <http://127.0.0.1:12000/preproc>

当用户选择完成预处理算法的序列之后，将序列数据传入这个接口。

传入数据的格式：

{

"data\_columns": [0, 1, 2, 3, 4, 5],

"pred\_column": -1,

"padding": ["mean", "median", "mean", "zero", "forestr", "forestr"],

"norm": ["zscore", "minmax", "zscore", null, null, null],

"sifting": {

"method": "if",

"th": null

}

}

data\_columns必须为一个数组，且非空，指示需要参与后续运算的列。

pred\_column必须为一个数（可以是负数表示倒数第几列，原则上只要能被Python的可迭代对象索引就可以），指示需要预测的目标列。

padding为一个字符串数组，指示对data\_columns中的每一列应用什么缺失值填补算法。如果不做处理，那么对应列的位置可以置null。如果padding本身为null,那么跳过该步骤。

padding中的算法可以是：'zero', 'mean', 'median',

'knnc', 'knnr', 'forestc', 'forestr'

norm为一个字符串数组，指示对data\_columns中的每一列应用什么缺失值填补算法。如果不做处理，那么对应列的位置可以置null。如果norm本身为null,那么跳过该步骤。

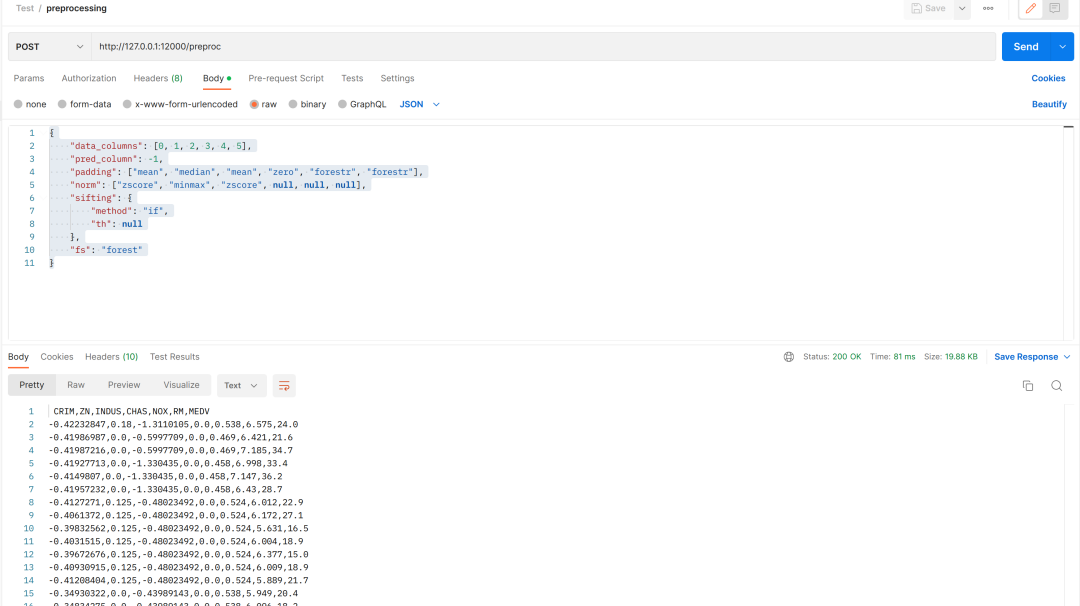
norm中的算法可以是：'minmax', 'zscore', 'sigmoid', 'log', 'l2'

padding和norm数组，如果有值，那么长度必须和data\_columns一致。

sifting用于传递数据筛选算法。th指定筛选的门槛。当method为if时，th可以不指定，或为一个浮点数；当method为dbscan时，th必须为浮点数。

sifting中的method可以取的值是：'dbscan', 'if'

该接口返回一个经过处理的csv文件。



1. <http://127.0.0.1:12000/fs>

该接口提供单变量参数相关性统计的算法。请求格式如下：

{

"method": "pearson"

}

method为pearson或forest。返回一张已经画好的直方图。

