接口调用测试与说明

本文档将带入具体的数据进行测试，并解释一部分的参数和逻辑，并会给出hdfs的测试方法

用户注册与登录

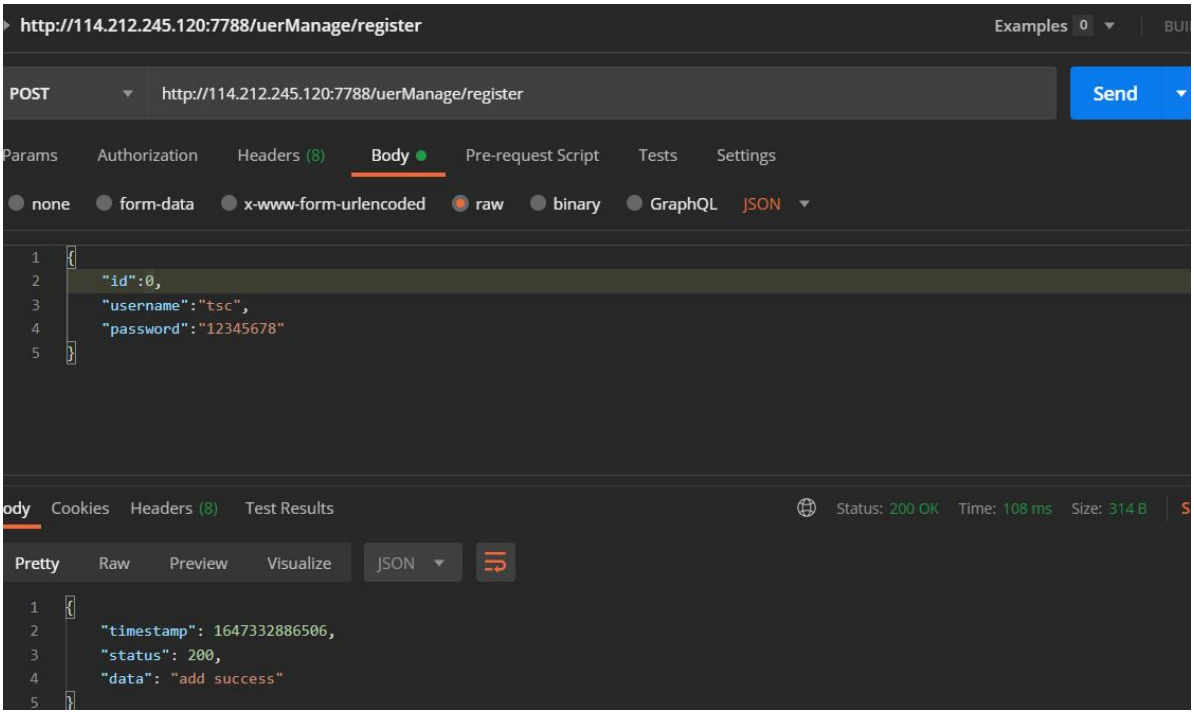
数据库中，已注册用户tsc，密码为12345678

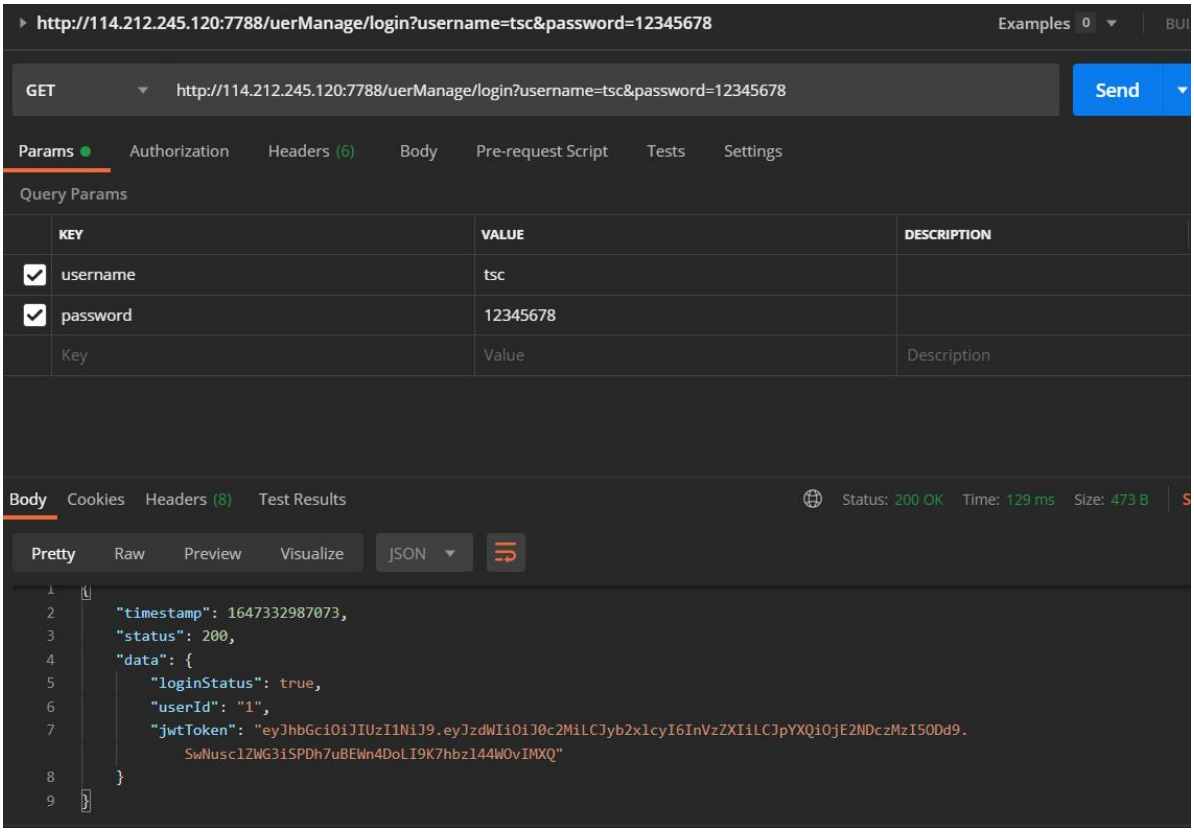
注册时id可以**随意取值**，因为数据库会自动填充id

请求成功后，用navicat打开user表，发现出现了tsc用户

如果想重置自动递增的id，用truncate table user

登录成功后，会返回userId，需要本地缓存，用于其他接口

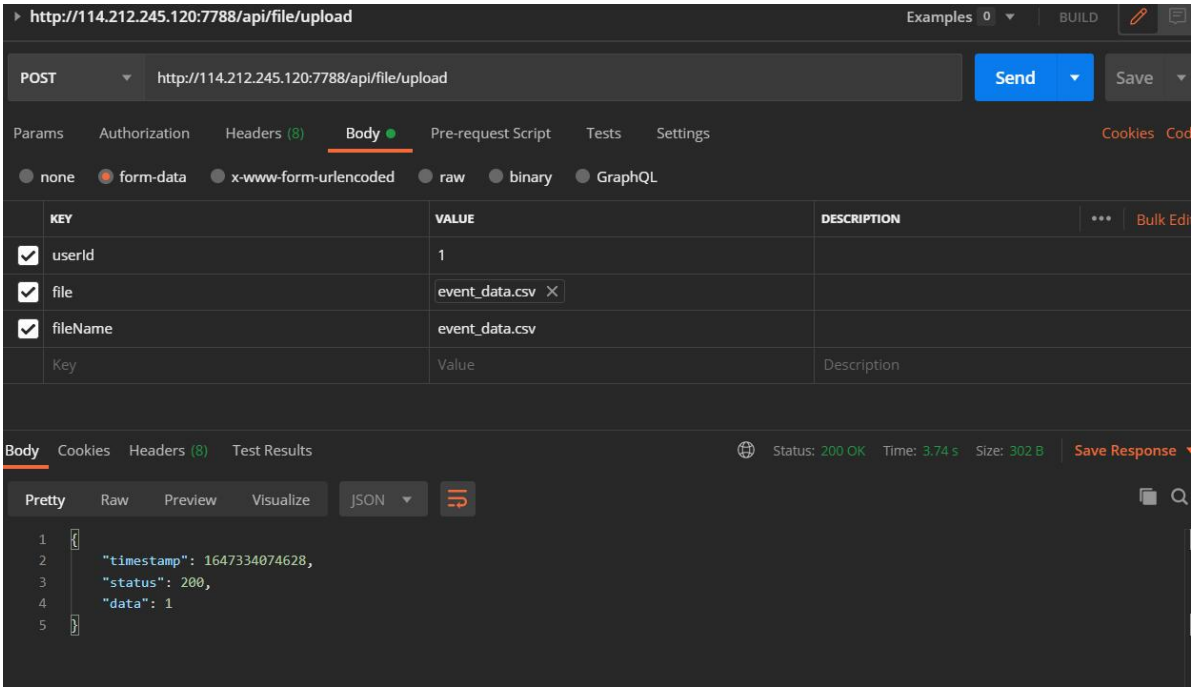




文件上传

注意：该接口没有判重，所以尽可能不要重复上传同一文件

如果要重新上传，先用hdfs命令删除



在postman里，点击file右侧可以选择text类型或file类型，file类型可以选择文件

上传成功后

hdfs操作相关方法

hdfs页面在50070端口

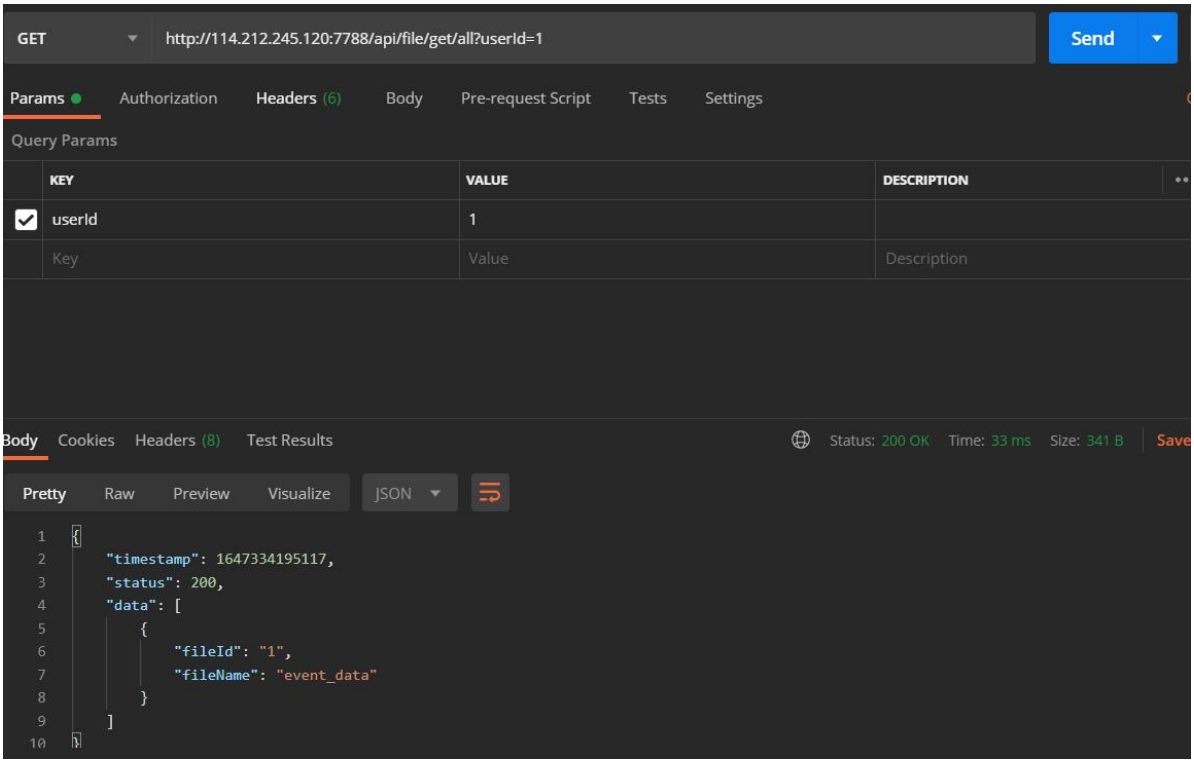
spark页面在8080端口

```
cd /ai_environment_jars/hadoop-2.7.4
#进入hadoop目录
bin/hdfs dfs -ls /data/java_compile/
#查看上传文件
bin/hdfs dfs -rm -r /data/java_compile/*
#删除所有文件
```

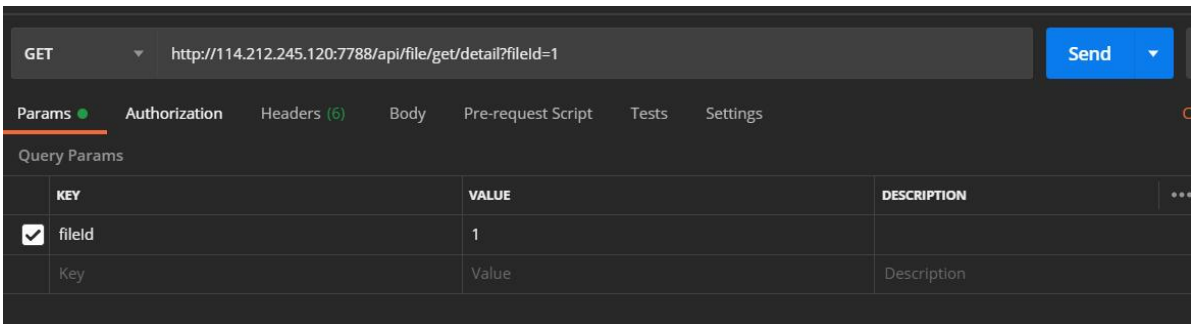
```
(base) seecoder@seecoder-System-Product-Name:~/ai_environment_jars/hadoop-2.7.4$ bi
n/hdfs dfs -ls /data/java_compile
Found 3 items
drwxr-xr-x   - seecoder supergroup          0 2022-03-12 17:30 /data/java_compile/e
vent_data
drwxr-xr-x   - seecoder supergroup          0 2022-03-12 20:15 /data/java_compile/e
vent_data123.libsvm
drwxr-xr-x   - seecoder supergroup          0 2022-03-12 17:35 /data/java_compile/e
vent_test
```

文件查询

getall只需要传入userId，这里传入tsc用户的id



getDetail接口传入文件id，这里传入上面upload的event_data.csv的fileId=1



结果为


```
{
    "timestamp": 1648474526058,
    "status": 200,
    "data": {
        "fileName": "event",
        "createTime": "2022-03-15T12:14:20.091+00:00",
        "fileStrucInfo": {
            "totalRows": "754"
        },
        "headerInfos": [
            {
                "id": 81,
                "fileInfoId": 1,
                "fieldName": "frame_index",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 1,
                "fieldType": "int",
                "valueInfo": "[\"79\",\"89992\"]"
            },
            {
                "id": 82,
                "fileInfoId": 1,
                "fieldName": "time",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 1,
                "fieldType": "double",
                "valueInfo": "[\"3.16\",\"3599.68\"]"
            },
            {
                "id": 83,
                "fileInfoId": 1,
                "fieldName": "road",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 0,
                "fieldType": "string",
                "valueInfo": "[\"R\",\"L\"]"
            },
            {
                "id": 84,
                "fileInfoId": 1,
                "fieldName": "lane",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 0,
                "fieldType": "string",
                "valueInfo": "[\"r\",\"m\",\"l\"]"
            },
            {
```

```
                "id": 85,
                "fileInfoId": 1,
                "fieldName": "vehicle_class",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 0,
                "fieldType": "string",
                "valueInfo": "[\"car\",\"truck\",\"bus\"]"
            },
            {
                "id": 86,
                "fileInfoId": 1,
                "fieldName": "vehicle_id",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 1,
                "fieldType": "int",
                "valueInfo": "[\"20\",\"46862\"]"
            },
            {
                "id": 87,
                "fileInfoId": 1,
                "fieldName": "event_type",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 0,
                "fieldType": "string",
                "valueInfo": "[\"cruise\",\"overdrive\"]"
            }
        ]
    }
}
```


具体含义查看接口文档

get/allfileandfunc

相比get/all，会返回可用的数理分析函数信息

这里的funcInfos是手动导入的，**有些参数作用不明**


```
{
    "timestamp": 1647335129081,
    "status": 200,
    "data": {
        "fileNamePojos": [
            {
                "fileId": "1",
                "fileName": "event_data"
            }
        ],
        "funcInfos": [
            {
                "id": 1,
                "funcDes": null,
```

```
                "funcName": "数据分布",
                "flagOfType": 2,
                "flagOfConOrDis": 2
            },
            {
                "id": 2,
                "funcDes": null,
                "funcName": "散点图",
                "flagOfType": 2,
                "flagOfConOrDis": 2
            },
            {
                "id": 3,
                "funcDes": null,
                "funcName": "数据统计",
                "flagOfType": 2,
                "flagOfConOrDis": 2
            },
            {
                "id": 4,
                "funcDes": null,
                "funcName": "偏度-峰度",
                "flagOfType": 2,
                "flagOfConOrDis": 2
            },
            {
                "id": 5,
                "funcDes": null,
                "funcName": "皮尔森相关系数",
                "flagOfType": 2,
                "flagOfConOrDis": 2
            },
            {
                "id": 6,
                "funcDes": null,
                "funcName": "平均值",
                "flagOfType": 2,
                "flagOfConOrDis": 2
            }
        ]
    }
}
```


文件分析

传入fileId=1,即上传的event_data.csv,结果为


```
{
    "timestamp": 1648474489138,
    "status": 200,
    "data": {
        "fileName": "event",
        "createTime": "2022-03-15T12:14:20.091+00:00",
        "fileStrucInfo": {
            "totalRows": "754"
        },
        "headerInfos": [
```

```
            {
                "id": 81,
                "fileInfoId": 1,
                "fieldName": "frame_index",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 1,
                "fieldType": "int",
                "valueInfo": "[\"79\",\"89992\"]"
            },
            {
                "id": 82,
                "fileInfoId": 1,
                "fieldName": "time",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 1,
                "fieldType": "double",
                "valueInfo": "[\"3.16\",\"3599.68\"]"
            },
            {
                "id": 83,
                "fileInfoId": 1,
                "fieldName": "road",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 0,
                "fieldType": "string",
                "valueInfo": "[\"R\",\"L\"]"
            },
            {
                "id": 84,
                "fileInfoId": 1,
                "fieldName": "lane",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 0,
                "fieldType": "string",
                "valueInfo": "[\"r\",\"m\",\"l\"]"
            },
            {
                "id": 85,
                "fileInfoId": 1,
                "fieldName": "vehicle_class",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 0,
                "fieldType": "string",
                "valueInfo": "[\"car\",\"truck\",\"bus\"]"
            },
            {
                "id": 86,
                "fileInfoId": 1,
```

```
                "fieldName": "vehicle_id",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 1,
                "fieldType": "int",
                "valueInfo": "[\"20\",\"46862\"]"
            },
            {
                "id": 87,
                "fileInfoId": 1,
                "fieldName": "event_type",
                "aliasName": "",
                "fieldDes": "",
                "nullsRatio": 0.0,
                "conOrDis": 0,
                "fieldType": "string",
                "valueInfo": "[\"cruise\",\"overdrive\"]"
            }
        ]
    }
}
```


当调用getDetail时似乎已经完成了分析过程

这里head_info字段id不同是因为想在数据库中吧自动递增设为1失败了，所以继续递增

模型类别

使用时可以通过GET：/get/models，通过模型类型的名称获得详细数据

参数的含义详见模型类别表model_type

id	类型名	中文名	参数列表
	LogisticRegression	逻辑回归	[MaxIter,RegParam,ElasticNetParam]
	DecisionTree	决策树	[Category,TrainingDataSetOccupy]
	RandomForest	随机森林	[MaxIter,RegParam,ElasticNetParam]
	GBDT	梯度提升决策树	[MaxIter,RegParam,ElasticNetParam]
	MultilayerPerceptronClassifier	多层感知器分类器	[Category,TrainingDataSetOccupy]
	NaiveBayes	朴素贝叶斯	[Category,TrainingDataSetOccupy]
	RandomForestRegression	随机森林回归	[Category,TrainingDataSetOccupy]
	K-Means	K-均值	[NumOfCluster,Seed]

分析函数类别

目前分析函数表func_info其他属性意义不明

func_id	数据分布
	数据分析
	散点图
	数据统计
	偏度-峰度
	皮尔森相关系数
	平均值

文件数据分析

这里/analysis/fields1等价于/analysis/fields传入funcId=1

数理函数分析方法与funcId的对应见上**分析函数类别**

数理函数分析需要在int或double字段上计算，否则会报错，需要前端有报错信息

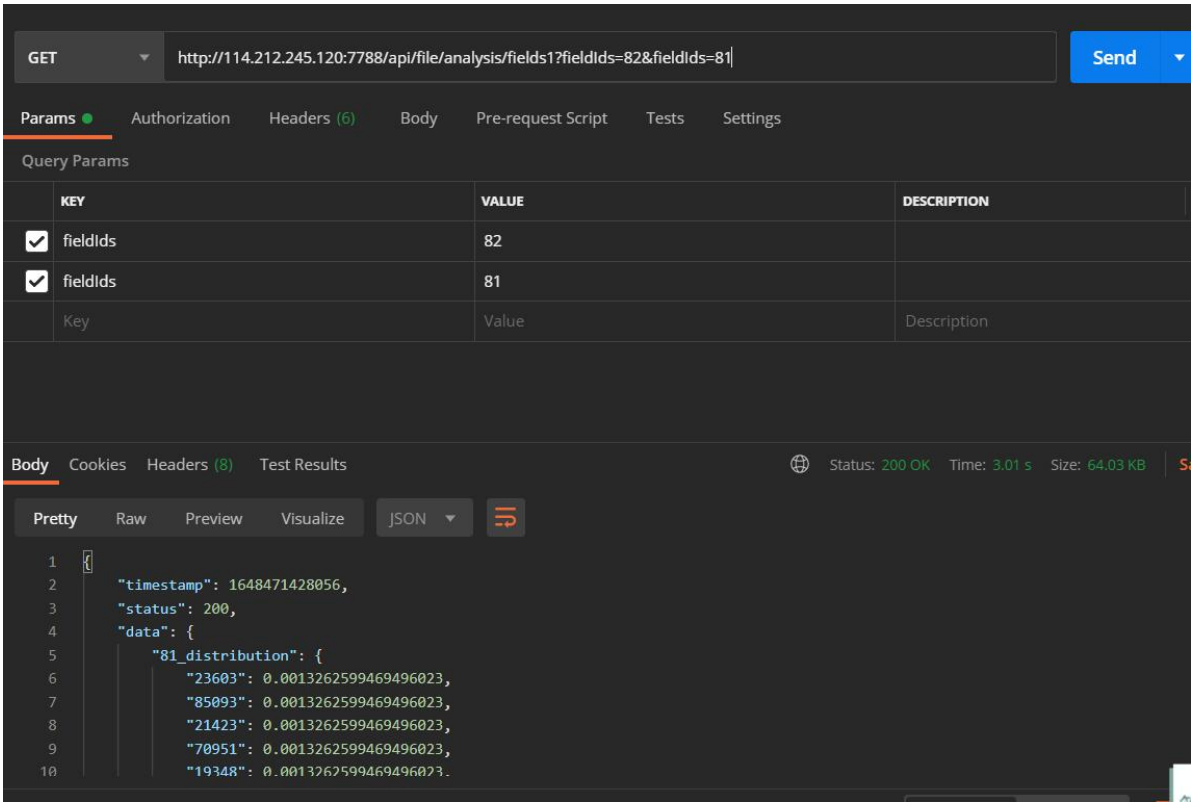
GET：/analysis/fields1

调用FileService.getFieldDistribution()进行文件数据分布分析

参数为fieldIds分析字段列表

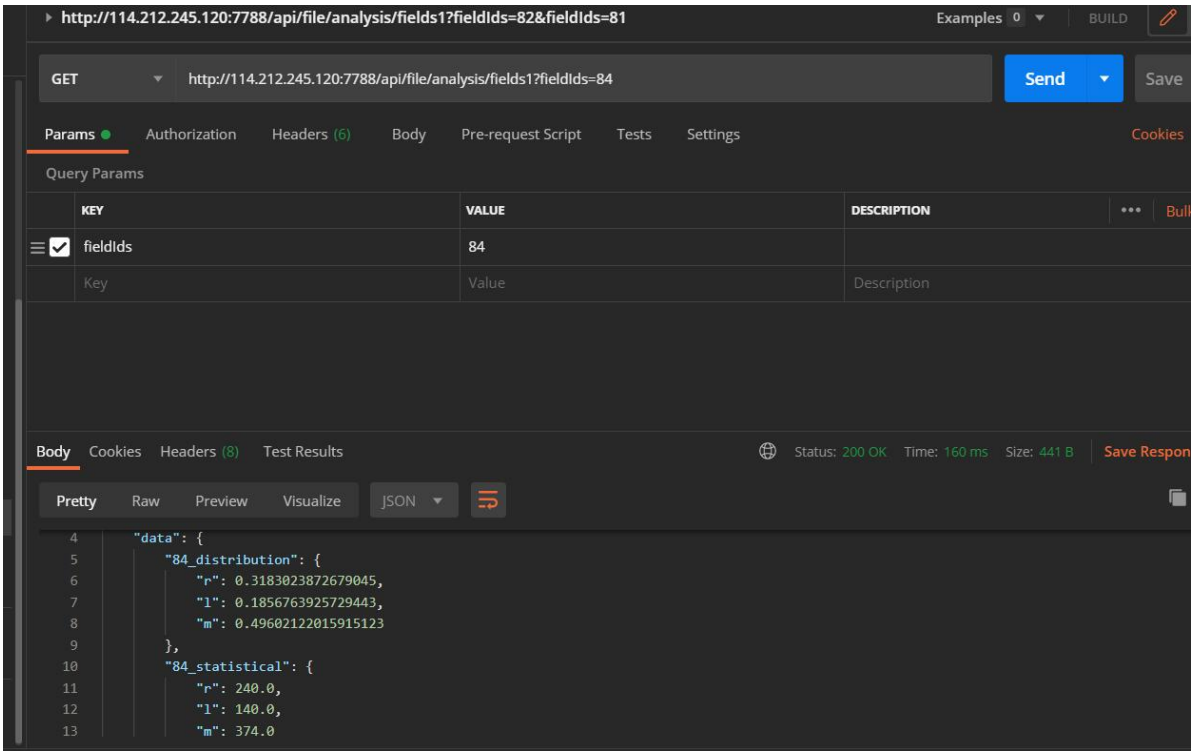
注意这里是fieldId而不是fileId，容易混淆，同时注意之前的java接口md有误，已更正

这里的0.0013...应该是字段值的频率



第二个更好理解的例子，选用车道占有率(l/m/r)字段

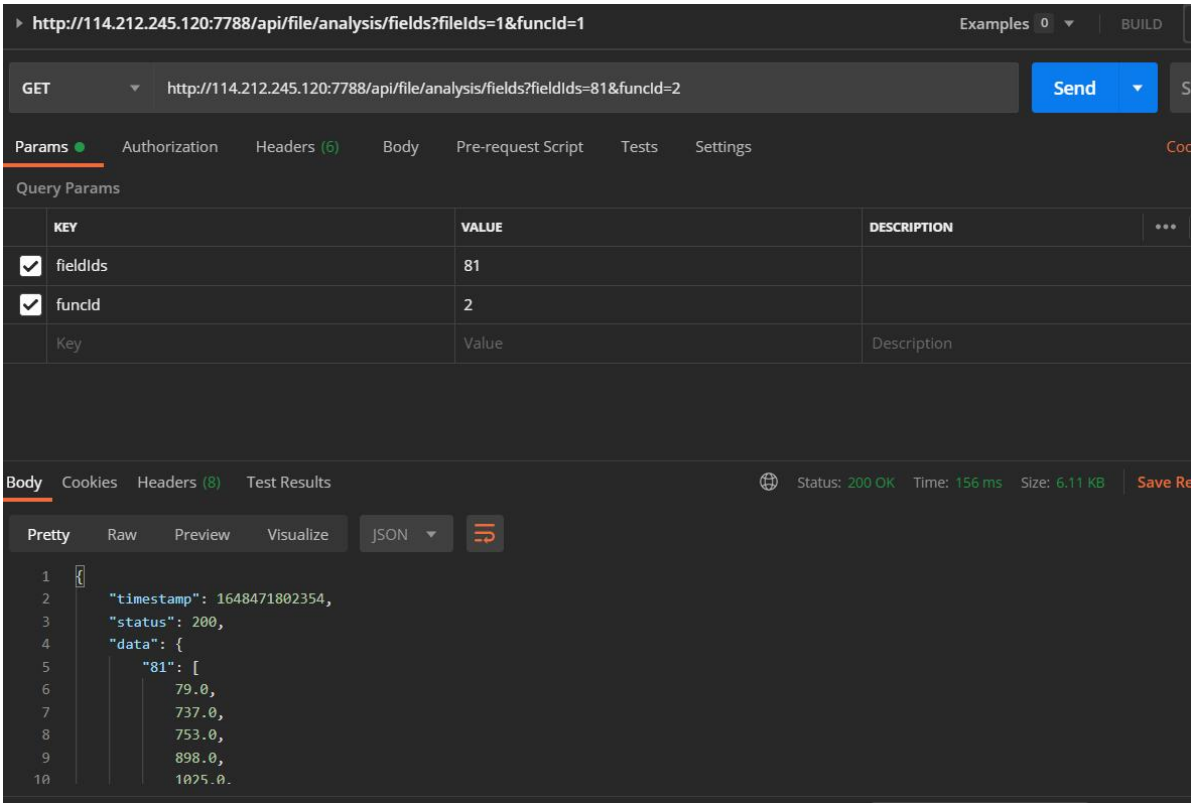
这里distreibution表示车道概率，statistical表示数量统计



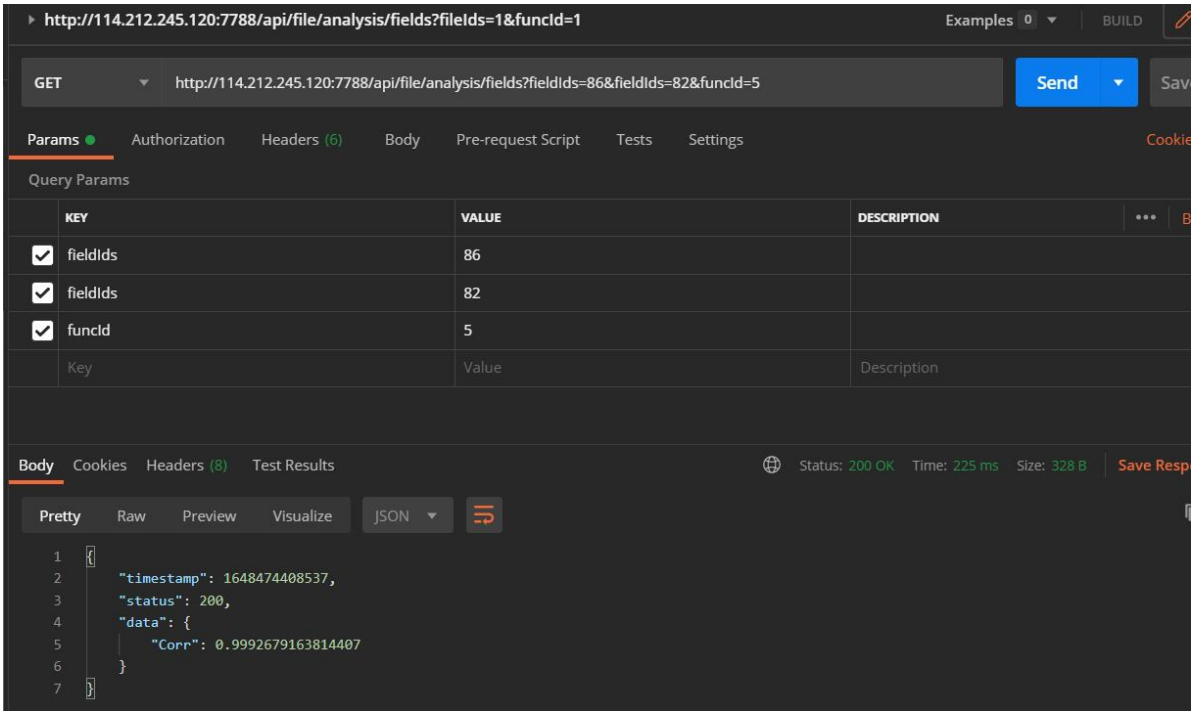
GET：/analysis/fields1

调用FileService.getFieldAnalysis()进行字段数理函数分析

这里是散点图分析



funcId=5时，即皮尔斯相关系数分析，需要至少两个类型为int或double的字段，否则无结果

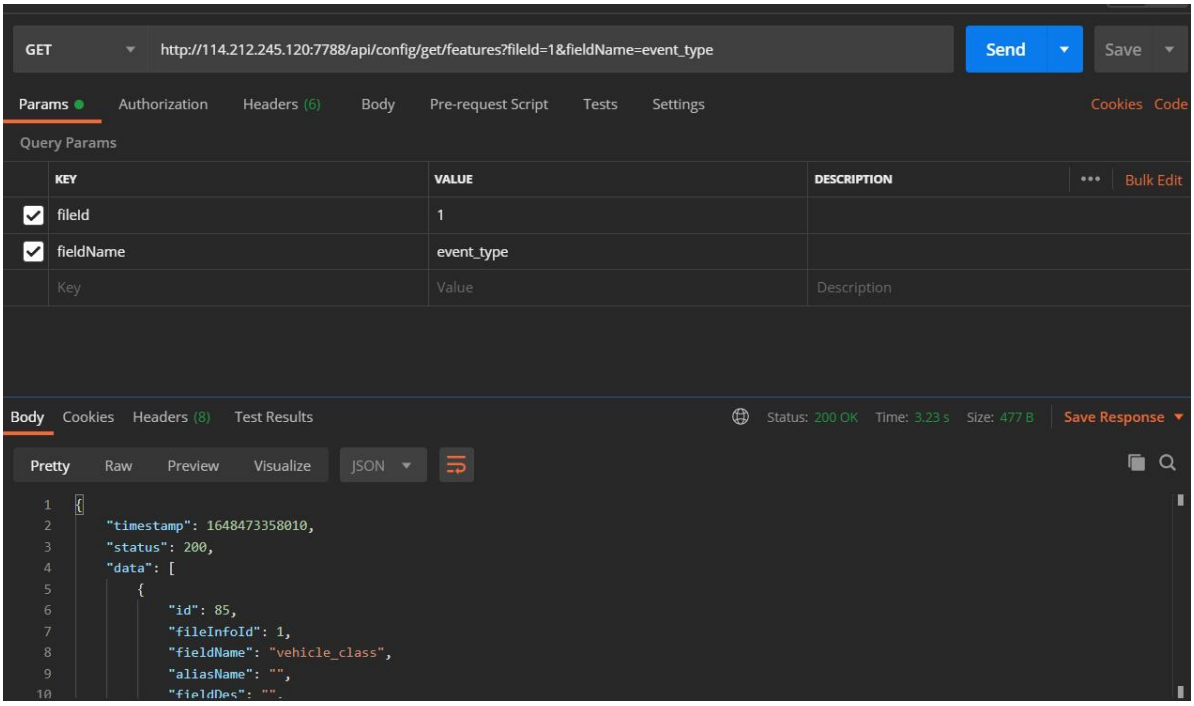


模型创建与使用相关

GET: /get/features

用户配置机器学习模型时进行自动化特征选择，然后查询这些自动化生成的特征信息

这里以高速监控为例，选择分析目标为event_type，返回结果vehicle_class,代表分析车辆类型与事故类型关系



GET：model/train 模型训练

这里使用的是modelId=1，model信息如下

```
{
    "id":"1",
    "modelTypeId":"2",
    "configId":"1",
    "fileInfoId":"1",
```

```
    "modelName":"event_data",
    "isTrained":false,
    "modelPath":"",
    "resOfModel":null,
    "arguments":{
        "Category":"10",
        "TrainingDataSetOccupy":"0.7"
    }
}
```

这里不需要的参数有：isTrained，modelPath，随便设置即可

config设置：

```
{
    "id":"0",
    "fileInfoId":"1",
    "userId":"1",
    "modelTypeId":"2",
    "confName":"config1",
    "fieldIds":["105","106","107","110","109"]
}
```

注意，这里字段Id与上面的不统一，因为这是新建的config

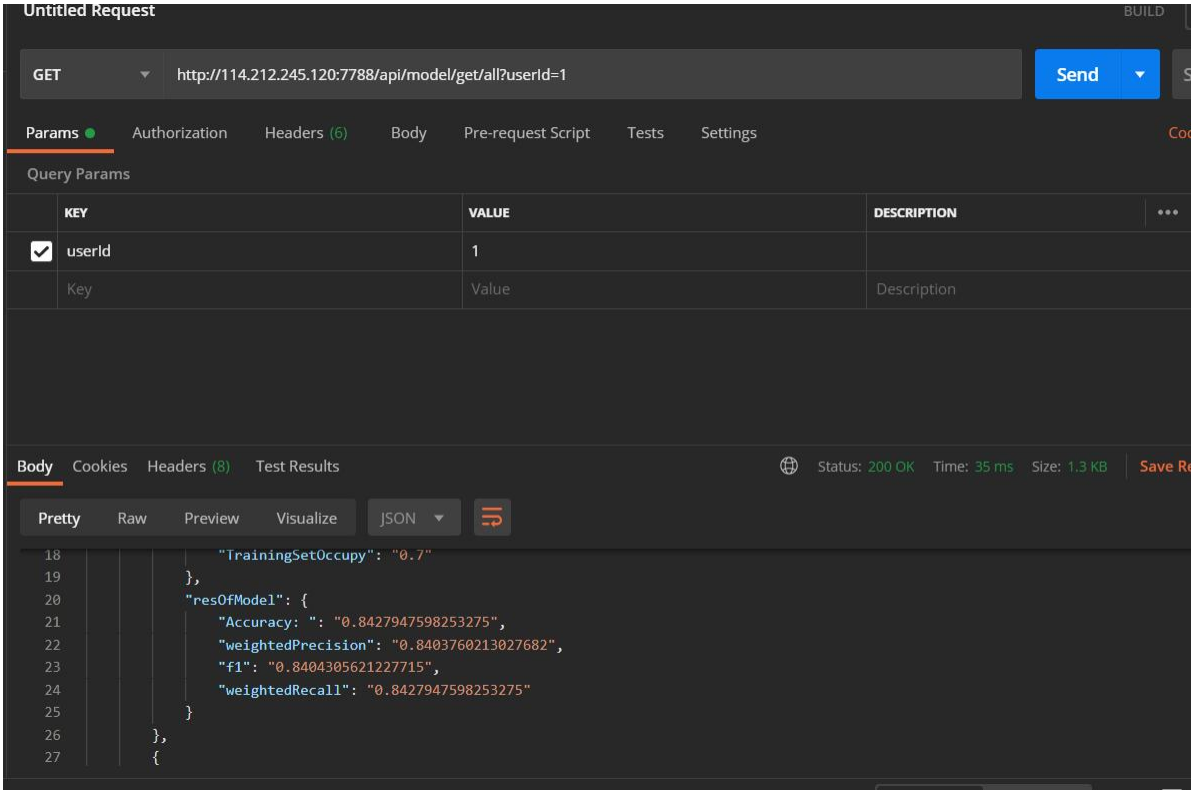
最后一个字段是目标字段，其他都是训练特征

即学习事故类型event_type与道路方向，车道，车型，时间的关系

id	alias_name	con_or_dis	field_des	field_name	field_type	file_info_id	nulls_ratio	value_info
103		1		frame_index	int	1	0	["79","89992"]
104		1		time	double	1	0	["3.16","3599.68"]
105		0		road	string	1	0	["R","L"]
106		0		lane	string	1	0	["r","m","l"]
107		0		vehicle_class	string	1	0	["car","truck","bus"
108		1		vehicle_id	int	1	0	["20","46862"]
109		0		event_type	string	1	0	["cruise","overdrive
110		0		duration	string	1	0	["48,60","24,36","36

modelPath在训练完成后，会将训练结果存储，将路径作为modelPath更新

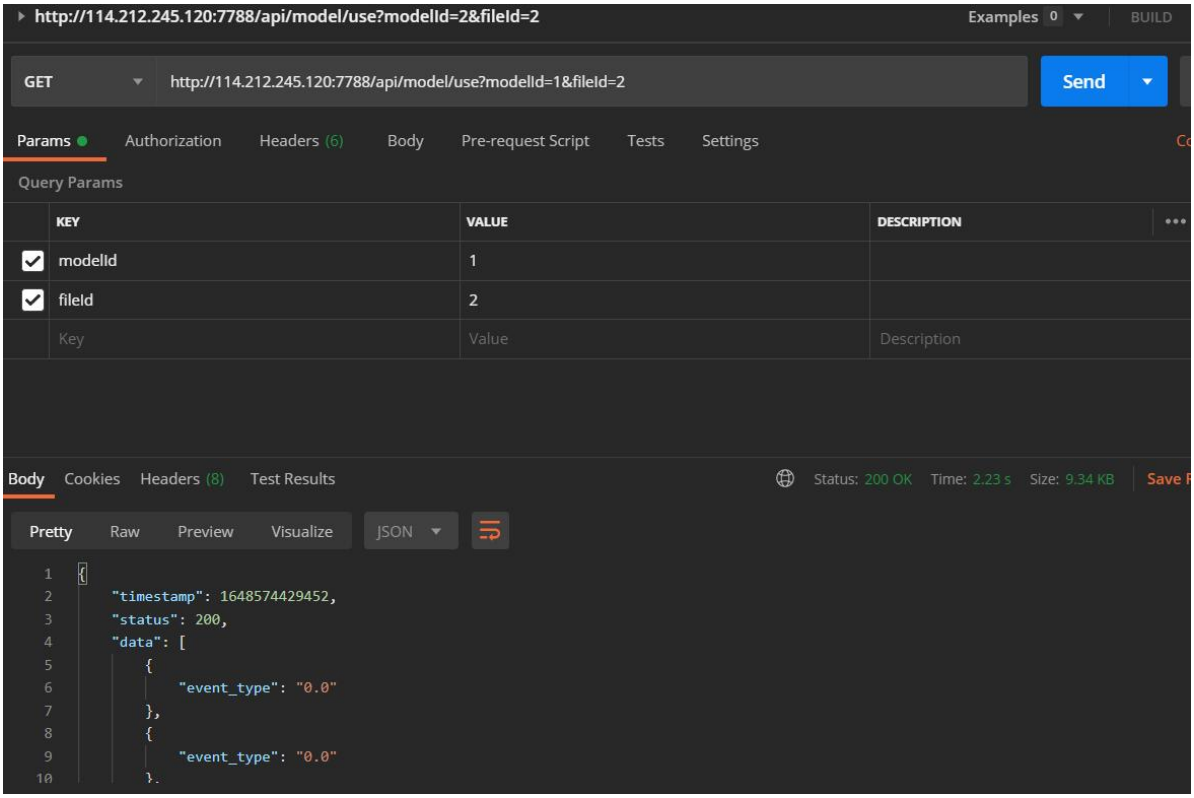
训练结果如下：



GET：api/model/use 模型使用

使用另一个结构相同的csv文件，文件信息如下

```
{
    "id":"2",
    "file_struc_info":{"totalRows":"441"},
    "file_name":"event_test",
    "folder":"",
"location":"hdfs://114.212.245.120:9000/data/java_compile/event_test",
    "upload_time":"2022-03-29 04:30:36.344000",
    "user_id":"1"
}
```



这里应该是吧事件类型离散了