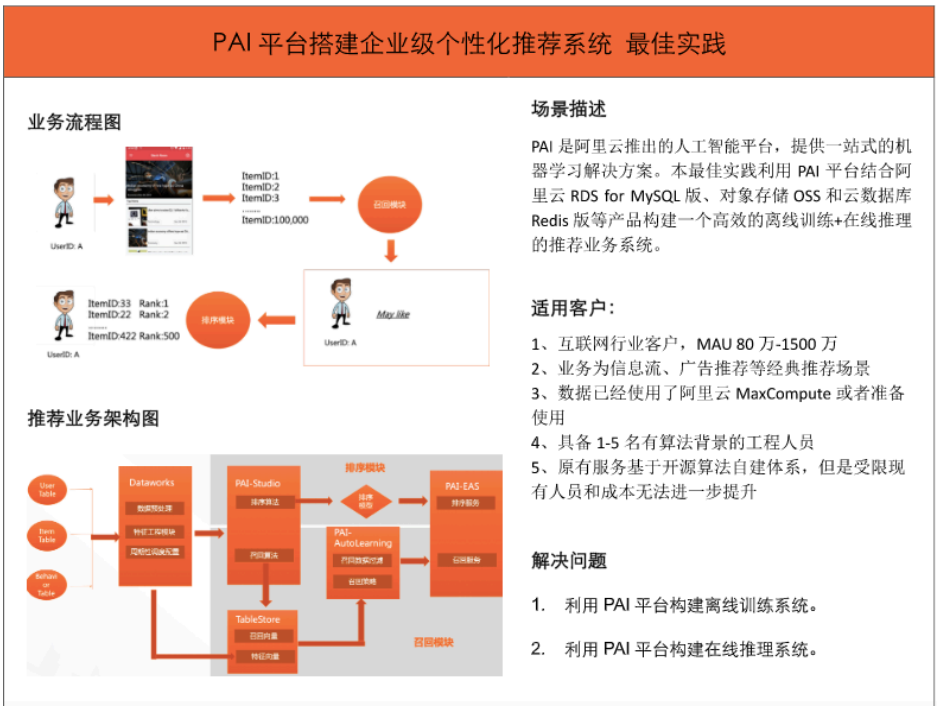
机器学习PAI

PAI最佳实践

PAI最佳实践

智能推荐解决方案

推荐业务端到端完整方案



140页超详细端到端实现文档如下：https://www.aliyun.com/acts/best-practice/preview?id=378791

ALS算法实现用户音乐打分预测

ALS算法实现用户音乐打分预测

很多人在决定是否看一部电影之前都会去豆瓣看下评分作为参考，看完电影也会给一个自己的分数。每个人对每个商品或者电影或是音乐都有一个心理的分数，这个分数标明用户是否对这个内容满意。作为内容的提供方，如果可以预测出每个用户对于内容的心理分数，就能更好的理解用户，并给用户提供好的内容推荐。今天就介绍下如何通过ALS矩阵分解算法实现用户对于音乐或者电影的评分预测。

ALS算法介绍

ALS算法是基于模型的推荐算法，基本思想是对稀疏矩阵进行模型分解，评估出缺失项的值，以此来得到一个基本的训练模型。然后依照此模型可以针对新的用户和物品数据进行评估。ALS是采用交替的最小二乘法来算出缺失项的，交替的最小二乘法是在最小二乘法的基础上发展而来的。

从协同过滤的分类来说，ALS算法属于User-Item CF，也叫做混合CF，它同时考虑了User和Item两个方面。

我们通过音乐打分这个案例介绍下交替最小二乘法的原理，首先拿到的原始数据是每个听众对每首歌的评分矩阵A，这个评分可能是非常稀疏的，因为不是每个用户都听过所有的歌，也不是每个用户都会对每首歌评分。

	痴心绝对	小酒窝	红豆	明天你好	浮夸
听众1	5			4	
听众2		6			3
听众3	3		7		
听众4				4	
听众5		4			6

ALS矩阵分解会把矩阵A分解成两个矩阵的相乘，分别是X矩阵和Y矩阵，

```
矩阵A=矩阵X和矩阵Y的转秩的乘积
```

x的列表示和Y的横表示可以称之为ALS中的因子，这个因子是有隐含定义的，这里假设有3个因子，分别是性格、教育程度、爱好。A矩阵经过ALS分解出的X、Y矩阵可以分别表示成：

	性格	教育程度	兴趣爱好
听众1	X_{11}	X_{12}	X_{13}
听众2	X_{21}	X_{22}	X_{23}
听众3	X_{31}	X_{32}	X_{33}
听众4	X_{41}	X_{42}	X_{43}
听众5	X_{51}	X_{52}	X_{53}

(上图为x矩阵)

	痴心绝对	小酒窝	红豆	明天你好	浮夸
性格	Y_{11}	Y_{12}	Y_{13}	Y_{14}	Y_{15}
教育程度	Y_{21}	Y_{22}	Y_{23}	Y_{24}	Y_{25}
兴趣爱好	Y_{31}	Y_{32}	Y_{33}	Y_{34}	Y_{35}

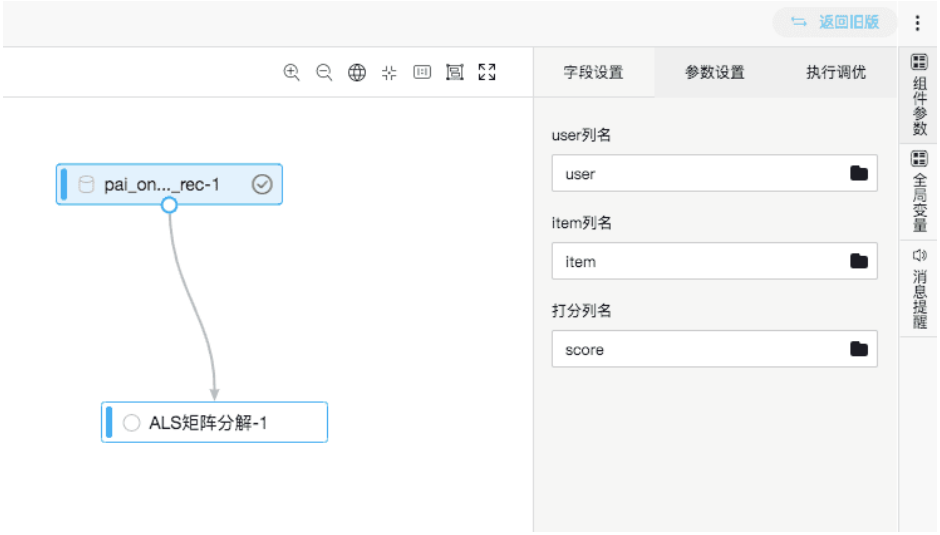
(上图为Y矩阵)

数据经过这样的拆解就很容易做用户对音乐的评分预测。比如有听众6，他从没听过“红豆“这首歌，但是我们可以拿到听众6在矩阵分解中X矩阵的向量M，这时候只有把向量M和”红豆“在Y矩阵中的对应向量N相乘，就能预测出听众6对于”红豆“这首歌的评分。

ALS在PAI实验

现在在PAI上面对ALS算法案例进行实验。整体流程只需要包含输入数据源和ALS矩阵分解组件即可。本案例已经集成于PAI-STUDIO首页模板：

创建后如图：



1.数据源

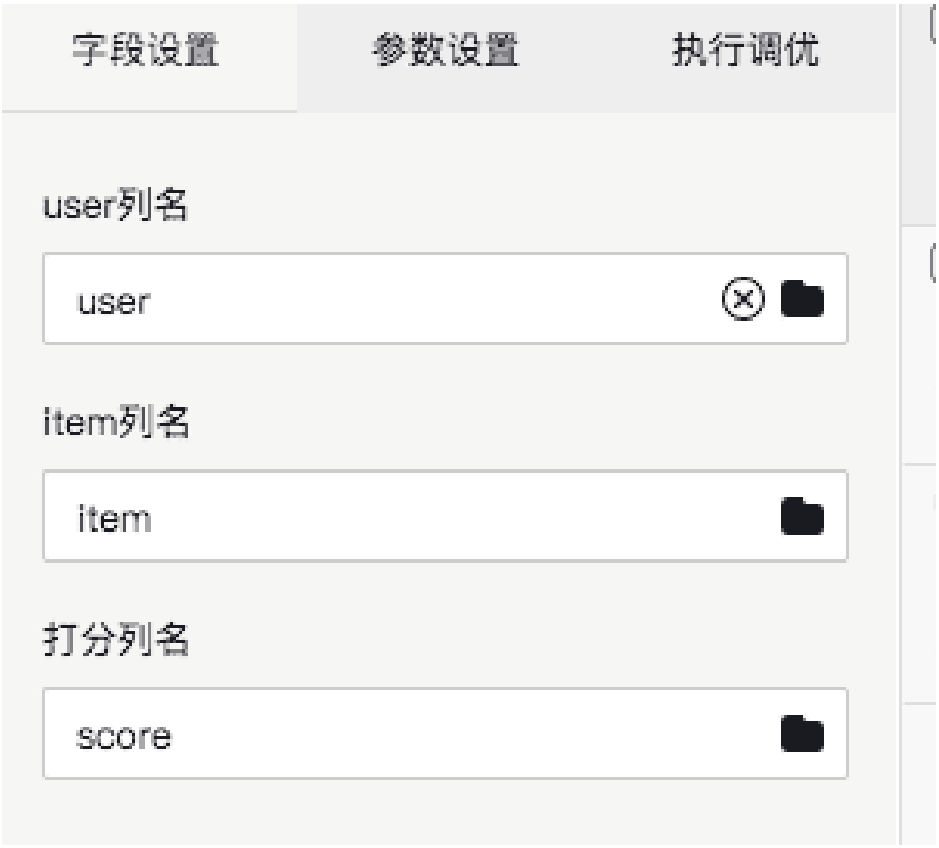
输入数据源包含4个字段

id	user	score	item
5	3249	1	978245916
5	3176	2	978243085
5	1719	3	978244205
5	2806	2	978243085
5	2734	2	978242788
5	1649	4	978244667
5	321	3	978245863

- User:用户ID
- Item：音乐ID
- score：user对item的评分

2.ALS矩阵分解

需要设置3个对应字段，



参数名称	参数描述	取值范围	是否必选，默认值
userColName	user列名	列的类型必须是bigint，可以不连续编号	必选
itemColName	item列名	列的类型必须是bigint，可以不连续编号	必选
rateColName	打分列名	列的类型必须是数值类型	必选
numFactors	因子数	正整数	可选，默认值100
numIter	迭代数	正整数	可选，默认值10
lambda	正则化系数	浮点数	可选，默认值0.1
implicitPref	是否采用隐式偏好模型	布尔型	可选，默认值false
alpha	隐式偏好系数	浮点数，大于0	可选，默认值40

3.结果分析

本案例中会输出2张表，对应ALS算法介绍中说的X矩阵和Y矩阵。

X矩阵表如图：

user ▲	factors ▲
1	[-0.14220297,0.8327106,0.5352268,0.6336995,1.2326205,0.7112976,0.9794858,0.8489773,0.330319,0.7426911]
2	[0.7714355,0.8170629,0.14070371,0.78157544,0.40145266,0.22435305,0.5998539,0.87861717,0.9321072,0.60098845]
3	[0.06963833,0.37125903,0.66982716,0.2325376,0.036257666,0.58954036,0.65054536,0.024004433,0.0033932994,0.57789034]
4	[0.64207155,0.8115232,0.32260254,0.3855561,0.25163174,0.40492404,0.5162408,0.3814767,0.67290497,0.50865084]
5	[0.517571,0.48458508,0.098304495,0.16832124,0.9891444,0.6789138,1.0585984,0.92578393,0.81489587,0.69474304]
6	[0.86565155,0.52865344,0.51986974,0.39816418,0.5968873,0.31424767,0.74578124,0.6733258,0.55831975,0.5425565]
7	[0.4147453,-0.27837437,0.4839715,0.7758234,0.6311068,0.84274673,0.4438908,0.8602465,0.3978993,1.4290581]
10	[0.47920293,0.91401875,0.95837015,0.7224187,0.5349992,0.7437093,0.33653644,1.0294899,0.4823215,0.41025826]
11	[0.54607016,0.23469958,0.32390735,0.5483177,0.07322444,0.87607765,0.25690663,0.75714564,0.19066288,0.2303486]

Y矩阵表如图：

item ▲	factors ▲
1009669227	[0.3043724,0.9211403,0.9649405,1.0043586,0.2320434,0.21626948,0.54844594,-0.3672228,0.09937295,0.9076632]
1009669181	[0.7098306,1.0229378,0.39896926,0.21804416,0.59587604,0.9355453,0.41796923,0.3523143,0.6874485,0.6521343]
1009669116	[0.3661423,0.3652928,0.8348509,0.9079304,0.7299789,0.2659982,0.26861745,0.65150297,0.6419628,1.2271518]
1009669115	[1.1625334,0.48568162,0.6818684,0.6328848,0.356604,-0.14263554,0.30305552,0.88706565,0.42701712,0.07457363]
1009669071	[0.39142805,0.06098657,0.3756292,1.0510693,0.42343494,0.86710936,0.4328914,0.09838692,-0.034022175,0.4868143]
994556636	[0.71699333,0.5847747,0.96564907,0.36637592,0.77271074,0.52454436,0.69028413,0.2341857,0.73444265,0.8352135]
994556598	[0.5234192,0.40755722,0.55578834,0.4585709,0.55235267,0.73103094,0.40249807,0.30472404,0.5356546,0.63388145]
993707035	[0.13577692,0.31376198,0.23644955,0.060735635,-0.083099656,0.16841954,0.1623567,0.21238364,0.18928273,0.123004556]
993707016	[0.1835768,0.74266636,0.49669686,0.2840153,0.8125185,0.36599895,0.31735852,0.31228343,0.9716536,0.11837222]
993706986	[0.171457,0.7812586,0.36249438,0.24480419,0.68455917,0.079008356,0.6320103,0.60387015,0.280187,0.38793203]

比如要预测user1对音乐item994556636的评分，只要将下方两个向量相乘即可

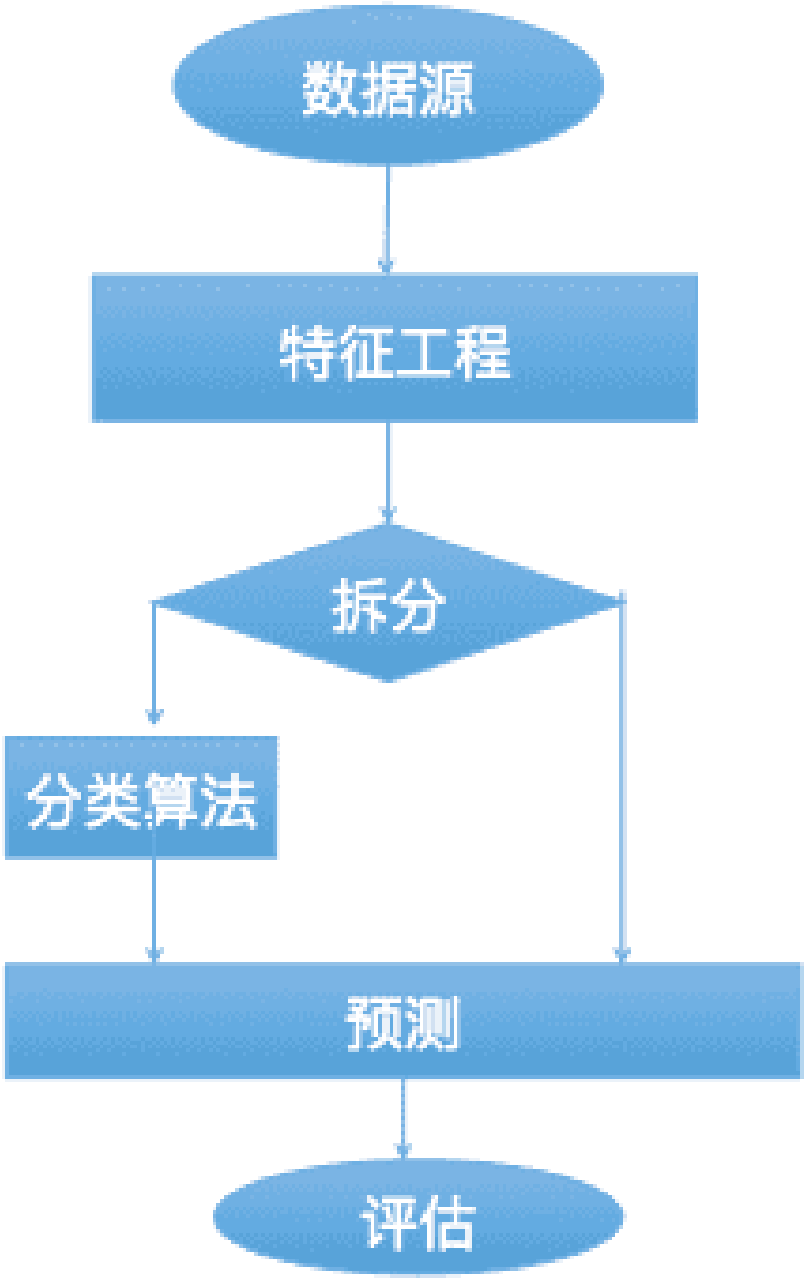
- User1：[-0.14220297,0.8327106,0.5352268,0.6336995,1.2326205,0.7112976,0.9794858,0.8489773,0.330319,0.7426911]
- item994556636：[0.71699333,0.5847747,0.96564907,0.36637592,0.77271074,0.52454436,0.69028413,0.2341857,0.73444265,0.8352135]

基于对象特征的推荐

（本实验选用数据为真实电商脱敏数据，仅用于学习，请勿商用）

在上一期基于协同过滤的的推荐场景中，我们介绍了如何通过PAI快速搭建一个基于协同过滤方案的推荐系统，这一节会介绍一些如何基于推荐对象特征的推荐方法。

首先看下整个业务流程图，这是一个基于对象特征的推荐场景的通用流程：



- 首先把数据导入Maxcompute，有监督的结构化数据
- 接着做特征工程，在特征工程环节主要做一些数据的预处理以及特征的衍生，特征衍生的作用是扩充数据维度，使得数据能更大限度的表示业务特点
- 接着把数据通过拆分分成两份，一份通过分类算法生成二分类模型，另一份数据对模型效果进行测试
- 最后通过评估组件得到模型效果

一、业务场景描述

通过一份真实的电商数据的4、5月份做模型训练生成预测模型，通过6月份的购物数据对预测模型进行评估最终选择最优的模型部署为在线http服务供业务方调用。

本次实验选用的是PAI-Studio作为实验平台，仅通过拖拽组件就可以快速实现一套基于对象特征的推荐系统。本实验的数据和完整业务流程已经内置在了PAI首页模板，开箱即用：

二、数据集介绍

数据源：本数据源为天池大赛提供数据，数据按时间分为两份，分别是7月份之前的购买行为数据和7月份之后的。具体字段如下：

字段名	含义	类型	描述
user_id	用户编号	string	购物的用户ID
item_id	物品编号	string	被购买物品的编号
active_type	购物行为	string	0表示点击，1表示购买，2表示收藏，3表

			示购物车
active_date	购物时间	string	购物发生的时间

数据截图：

10944750	8689	2	5月2日
10944750	25687	2	5月8日
10944750	7150	1	6月7日
10944750	13451	0	6月4日
10944750	13451	0	6月4日
10944750	13451	0	6月4日
10944750	13451	0	6月4日
10944750	13451	0	6月4日

三、数据探索流程

本次实验选用的是PAI-Studio作为实验平台，仅通过拖拽组件就可以快速实现一套基于协同过滤的推荐系统，并且支持自动调参以及模型一键部署的服务。

实验流程图：

（1）特征工程

在特征工程的流程中是把最原始的只有4个字段的数据通过特种工程的方法进行数据维度的扩充。在推荐场景中有两个方面特征，一方面是所推荐的对象的特征，另一方面是被推荐对象的特征。

在商品推荐这个案例中：

- 被推荐对象为商品（item），扩充的维度为每个item被购买量、每个item被点击量、每个item被点击购买率（购买量除以点击率）
- 推荐对象为用户（user），扩充的维度为每个user总的购买量、总的点击量、总的点击购买率（点击数除以购买率，可以得出每点击多少次购买一个产品，可以用来描述用户购物的果断性）

最终数据由原始的4个字段变成了10个字段：

user_id	item_id	active_type	active_month
10944750	13451	0	6
10944750	13451	2	6
10944750	13451	2	6
10944750	13451	0	6
10944750	13451	0	6
10944750	13451	0	6
10944750	13451	0	6

item_id	user_id	active_type	active_month	item_total_buy	item_total_count	item_buy_rate	user_total_count	user_total_buy_count	user_buy_rate
1000	12016750	0	5	1	4	0.25	221	18	0.08144796380090498
1000	12016750	0	5	1	4	0.25	221	18	0.08144796380090498
1000	12016750	0	5	1	4	0.25	221	18	0.08144796380090498
1000	12016750	0	5	1	4	0.25	221	18	0.08144796380090498
10000	5901250	0	6	0	2	0	50	0	0
10000	5901250	0	6	0	2	0	50	0	0
10000	5901250	0	6	0	2	0	50	0	0
10000	5901250	0	6	0	2	0	50	0	0
10010	2921750	0	5	0	2	0	528	11	0.020833333333333332

（2）模型训练

现在已经构建了一个大宽表，有了做完特征工程的结构化数据，现在就可以训练模型了。这个案例中选用了逻辑回归算法，在做模型训练过程中有一个痛点就是如何找到合适的参数，对于逻辑回归参数（如下图）而言，如何调整以下几个参数，使得模型训练能达到最好的效果是一个非常有挑战的任务。

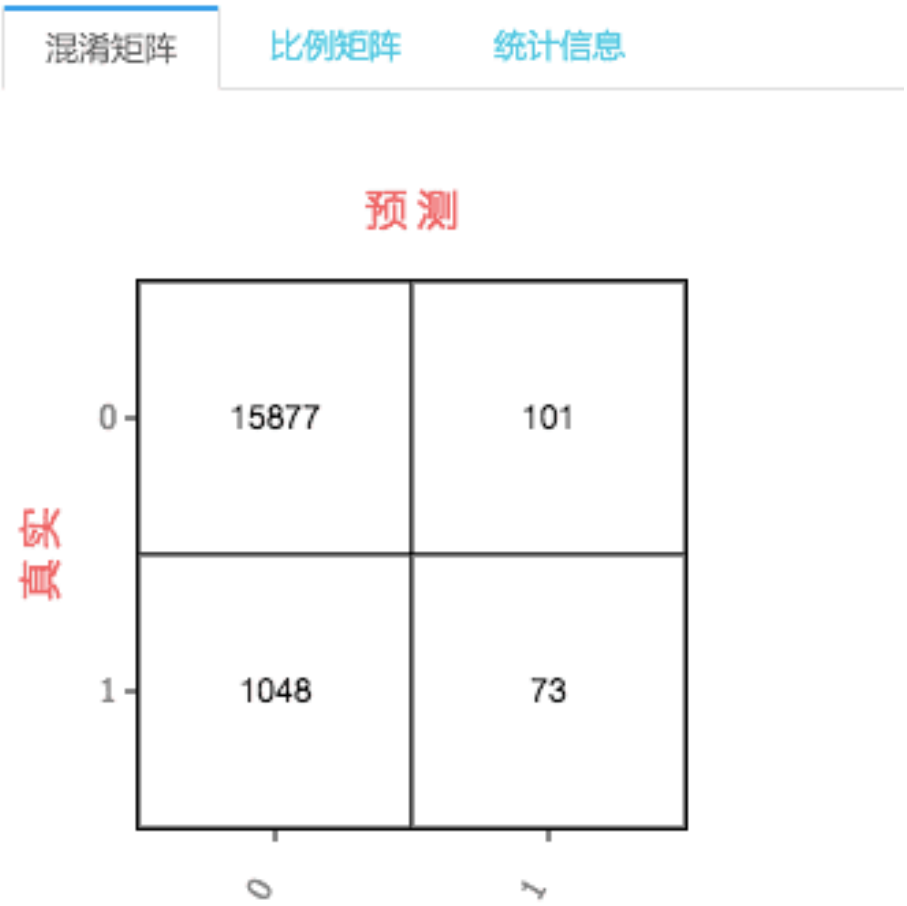
为了解决繁琐的调参工作带来的劳动量问题，PAI产品内置了AutoML引擎帮助调参，在页面上打开AutoML，只要设置下需要调参的算法的参数范围以及评估标准，后台引擎即可在最小的资源消耗下找到最合理的参数，详见:

（3）模型评估

模型评估模块是用预留的一部分未参与模型训练的数据评估模型质量，通常推荐场景都是二分类实验，可以使用混淆矩阵和二分类评估组件去评估结果。

二分类评估：打开组件选择“图表”，会展示下图ROC曲线，其中蓝色区域的面积为AUC值，面积越大表示模型质量越高

混淆矩阵：通过混淆矩阵可以确定具体的预测准确率、召回率、F1-Score等指标



（4）模型在线部署

模型生成后，如果效果也达到预期，可以使用PAI-EAS将模型一键部署为在线服务，通过http访问。点击画布上的“部署”按钮，选择“模型在线部署”功能，选择需要部署的模型。

后续流程可以参考在线预测文档：https://help.aliyun.com/document_detail/92917.html

部署成在线服务之后，模型服务可以通过http请求访问，这样就可以做到模型跟用户自身的业务结合，完成

PAI模型训练和业务应用的打通。

推荐召回场景FM-Embedding实现方案

背景

被实验案例的数据和完整实验流程已经内置于PAI-Studio建模平台https://data.aliyun.com/product/learn

进入PAI-Studio，首页模板最下方位置点击从模板创建“推荐场景-FM向量召回”开箱即用

智能推荐分为排序和召回两大模块，在召回模块中通常会采用将 用户User和待推荐的 内容Item 分别以向量表示，然后通过User和Item的向量乘积大小作为User对Item的感兴趣程度的判断。本案例介绍如何基于真实的推荐场景数据，通过使用PAI平台提供的FM算法和Embedding提取算法产生User和Item的描述向量。

详细流程

完整业务流程图：

1.数据说明

原始数据如图：

userid	age	gender	itemid	price	size	label
1	64	male	A	500	10	1
2	42	female	B	200	4	0
3	42	male	C	425	6	1
4	53	female	D	474	3	0
5	57	male	E	64	7	0
6	86	female	F	532	3	0
7	34	female	G	42	4	1
8	23	male	H	364	6	0
9	14	female	I	57	4	0
10	35	male	J	463	9	1

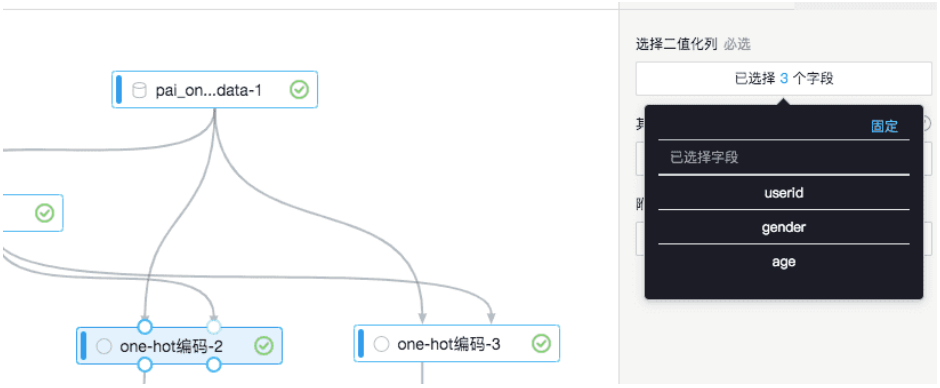
数据字段描述：

- userid：user的id信息
- age：user的年龄
- gender：user的年龄
- itemid：item的id信息
- price：item的价格
- size：item的大小
- label：目标列，是否购买，1为买，0为未买

2.One-hot编码

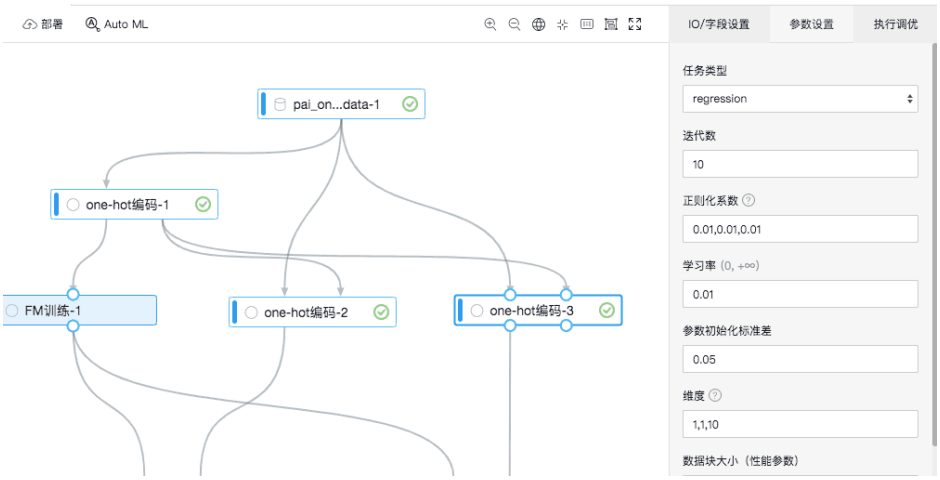
One-hot编码可以将字符型数据转成数值型表示，在FM-Embedding方案中首先利用“onehot编码-1”针对全量数据进行编码，生成编码模型再输入到“onehot编码-2”和“onehot编码-3”中，“onehot编码-2”需要选择User对应的特征信息进行编码，“onehot编码-3”选择Item对应的特征信息进行编码。

“onehot编码-2”的输入是userid、gender、age，附加列选择userid。



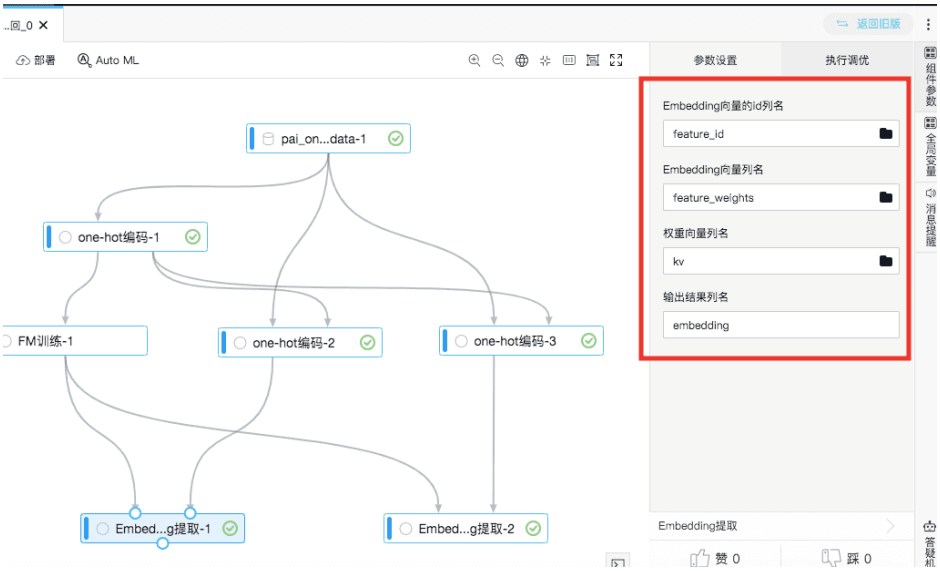
“onehot编码-3”的输入是itemid、price、size，附加列是itemid。

3.FM训练



正则化和维度这两个参数有3个参数，分别对应常数项、一次项和二次项。其中维度的第3个参数“10”代表生成的Embedding的维度。

4.Embedding提取



- Embedding向量id列名：输入左桩FM训练中的模型“feature_id”
- Embedding向量列名：输入左桩FM训练中的模型的“feature_weights”
- 权重向量列名：输入右桩对应的稀疏化数据列
- 输出结果列名：输出的Embedding字段名

最终输出结果：

userid	kv	embedding
1	10:1,7:1,37:1	0.04015407 −0.17816195 −0.037157465 −0.06470604 −0.24434555 −0.019216094 −0.049993407 −0.06353192 −0.08150465 0.001752859 0.3356...
2	9:1,4:1,39:1	−0.067233436 −0.13599731 0.12928867 −0.014686654 −0.079268694 −0.1312892 −0.092644565 0.027404211 0.00232377 −0.109620675 0.0445...
3	10:1,4:1,40:1	−0.004508253 −0.046913035 −0.07043892 0.010427853 −0.1450108 0.021560092 −0.10439287 0.055663645 −0.08991572 −0.014287934 0.440...
4	9:1,5:1,41:1	0.0050517395 −0.0021566674 −0.07513097 −0.10988943 0.031288043 −0.0033690166 −0.08820701 0.024628945 4.7708116E−4 0.048596375 −...
5	10:1,6:1,42:1	0.043785967 0.10553776 −0.19826782 −0.041631583 −0.01759258 0.021906495 −0.03562168 0.04236281 −0.12950923 −0.13433275 0.15293656
6	9:1,8:1,43:1	−0.076507404 −0.13286367 0.075596735 −0.039212134 0.14426178 0.025733178 −0.015803259 0.0065106675 −0.024862044 −0.12871072 −0.0...
7	9:1,2:1,44:1	−0.18068565 −0.096336134 0.037038583 −0.08846839 −0.0439286 0.015447946 −0.24221739 −0.08010515 −0.008318255 −0.05676799 0.1933...
8	10:1,1:1,45:1	0.052672688 −0.004056439 −0.09321347 −0.08363886 0.0086529665 0.01378352 −0.056089412 0.002947338 0.012545764 −0.036917157 0.02...
9	9:1,0:1,46:1	−0.06683848 −0.04957156 0.101151854 0.13750216 0.019501429 −0.0941189 −0.055305757 −0.02949195 0.067301184 −0.08456889 −0.045818195
10	10:1,3:1,38:1	−0.11435607 −0.076492555 −0.21123311 0.11723561 −0.15823722 0.011994862 0.02883054 −0.06578457 −0.1195012 0.05180212 0.5513177

总结

使用PAI提供的整套FM-Embedding方案可以在推荐业务场景中快速挖掘出User和Item对应的特征向量，在实际召回模块只要做User和Item的特征向量积就可以得到打分结果。

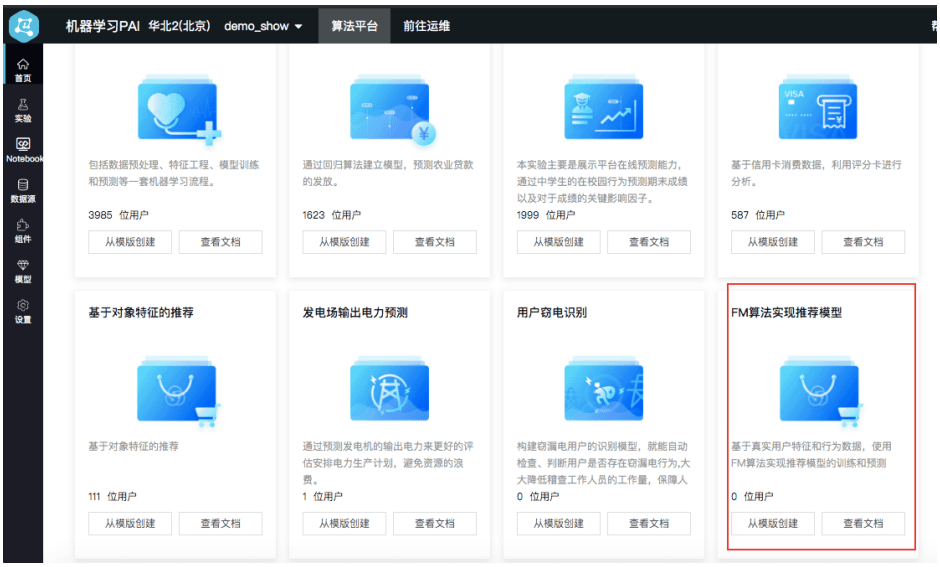
FM算法实现推荐模型

概述

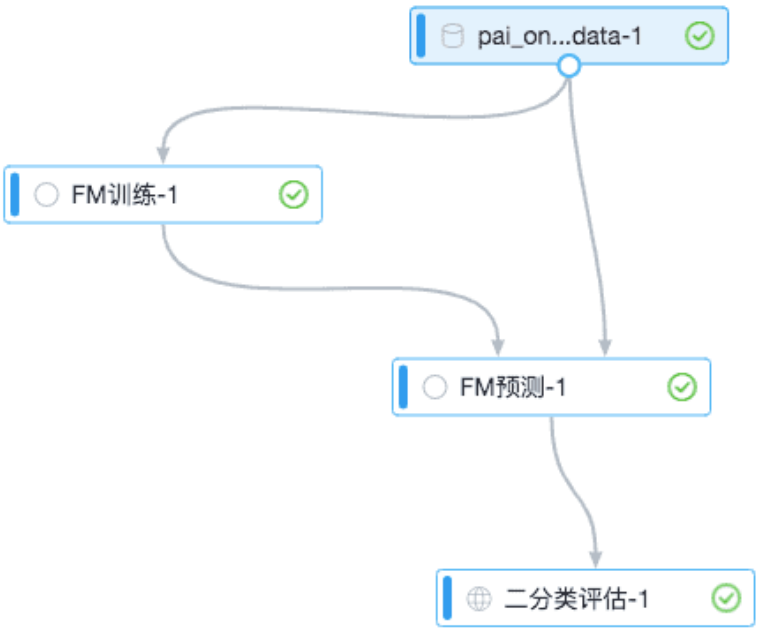
FM (Factorization Machine) 算法可进行回归和二分类预测，它的特点是考虑了特征之间的相互作用，是一种

非线性模型，目前FM算法是推荐领域被验证的效果较好的推荐方案之一，在诸多电商、广告、直播厂商的推荐领域有广泛应用。

PAI平台的FM算法基于阿里内部大数据的锤炼，具备性能优越、效果突出的特点。具体使用方式可以参见首页模板：



使用FM算法整体流程需要包含FM训练和FM预测组件，可以搭配评估组件使用。



输入数据要求

目前PAI的FM算法只支持libsvm格式的数据，数据需要包含两列，分别是特征列和目标列。

- 目标列：Double类型
- 特征列：String类型，特征要以k:v格式输入，特征直接以逗号分隔

如图：

序号	label	features
1	0	3:1,11:1,14:1,19:1,39:1,42:1,55:1,64:1,67:1,73:1,75:1,76:1,80:1,83:1
2	0	3:1,6:1,17:1,27:1,35:1,40:1,57:1,63:1,69:1,73:1,74:1,76:1,81:1,103:1
3	0	4:1,6:1,15:1,21:1,35:1,40:1,57:1,63:1,67:1,73:1,74:1,77:1,80:1,83:1
4	0	5:1,6:1,15:1,22:1,36:1,41:1,47:1,66:1,67:1,72:1,74:1,76:1,80:1,83:1
5	0	2:1,6:1,16:1,22:1,36:1,40:1,54:1,63:1,67:1,73:1,75:1,76:1,80:1,83:1
6	0	2:1,6:1,14:1,20:1,37:1,41:1,47:1,64:1,67:1,73:1,74:1,76:1,82:1,83:1
7	0	1:1,6:1,14:1,22:1,36:1,42:1,49:1,64:1,67:1,72:1,74:1,77:1,80:1,83:1
8	0	1:1,6:1,17:1,19:1,39:1,42:1,53:1,64:1,67:1,73:1,74:1,76:1,80:1,83:1
9	0	2:1,6:1,18:1,20:1,37:1,42:1,48:1,64:1,71:1,73:1,74:1,76:1,81:1,83:1
10	1	5:1,11:1,15:1,32:1,39:1,40:1,52:1,63:1,67:1,73:1,74:1,76:1,78:1,83:1

组件说明

1.FM训练

在“参数设置”中可以设置回归或者分类两种模式：

PAI命令

参数	解释	取值
tensorColName	训练的特征列名 (kv格式的字符串，例如”1:1.0,3:1.0”，特征的id必须是非负整数，取值范围是[0,Long.MAX_VALUE)，可以不连续）	必选

labelColName	label列名 (要求是数值类型，如果任务类型是binary_classification，那么label值必须是0或1)	必选
task	任务类型	必选，” regression” or “binary_classification”
numEpochs	迭代数	可选，默认值10
dim	因子数，字符串，用逗号分隔的三个整数，表示0次项、线性项、二次项的长度	可选，默认值 “1,1,10”
learnRate	学习率	可选， 默认值 0.01
lambda	正则化系数，字符串，用逗号分隔的三个浮点数，表示0次项、线性项、二次项的正则化系数	可选， 默认值 “0.01,0.01,0.01”
initStdev	参数初始化标准差	可选， 默认值0.05

备注1：

- 如遇到训练发散，可适当降低学习率的值

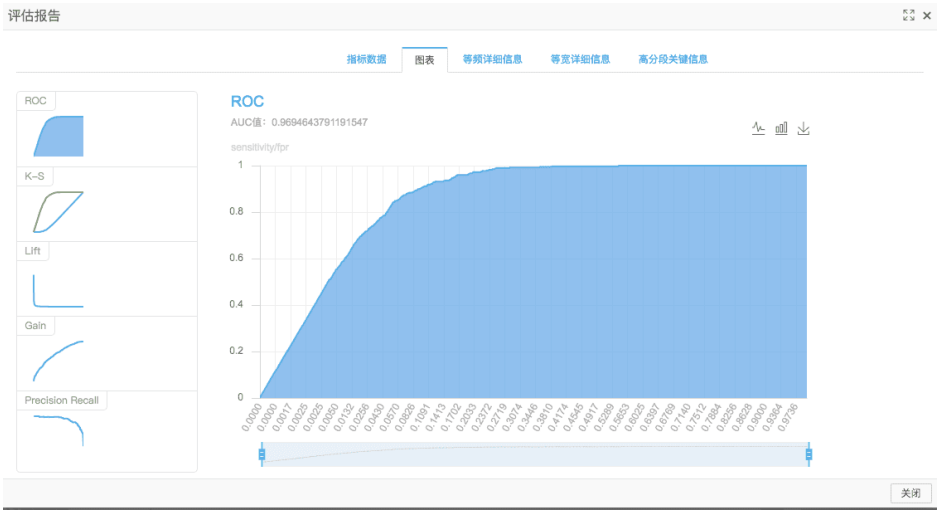
2.FM预测

PAI命令

参数	解释	取值
predResultColName	预测结果列名	可选，默认” prediction_result”
predScoreColName	预测得分列名	可选，默认” prediction_score”
predDetailColName	详细预测信息列名	可选，默认” prediction_detail”
keepColNames	保持到输出结果表的列	可选，默认全选

评估结果

在首页模板案例的数据情况下，使用PAI FM生成的模型可以达到接近0.97的AUC



协同过滤做商品推荐

本文数据为虚构，仅供实验。

背景

数据挖掘的一个经典案例就是尿布与啤酒的例子。尿布与啤酒看似毫不相关的两种产品，但是当超市将两种产品放到相邻货架销售的时候，会大大提高两者销量。很多时候看似不相关的两种产品，却会存在这某种神秘的隐含关系，获取这种关系将会对提高销售额起到推动作用，然而有时这种关联是很难通过理性的分析得到的。这时候我们需要借助数据挖掘中的常见算法-协同过滤来实现。这种算法可以帮助我们挖掘人与人以及商品与商品的关联关系。

协同过滤算法是一种基于关联规则的算法，以购物行为为例。假设有甲和乙两名用户，有a、b、c三款产品。如果甲和乙都购买了a和b这两种产品，我们可以假定甲和乙有近似的购物品味。当甲购买了产品c而乙还没有购买c的时候，我们就可以把c也推荐给乙。这是一种典型的user-based情况，就是以user的特性做为一种关联。

本文的业务场景如下：
通过一份7月份前的用户购物行为数据，获取商品的关联关系，对用户7月份之后的购买形成推荐，并评估结果。比如用户甲某在7月份之前买了商品A，商品A与B强相关，我们就在7月份之后推荐了商品B，并探查这次推荐是否命中。

数据集介绍

本文档数据源为天池大赛提供数据，数据按时间分为两份，分别是7月份之前和7月份之后的购买行为数据。

具体字段如下表。

字段名	含义	类型	描述

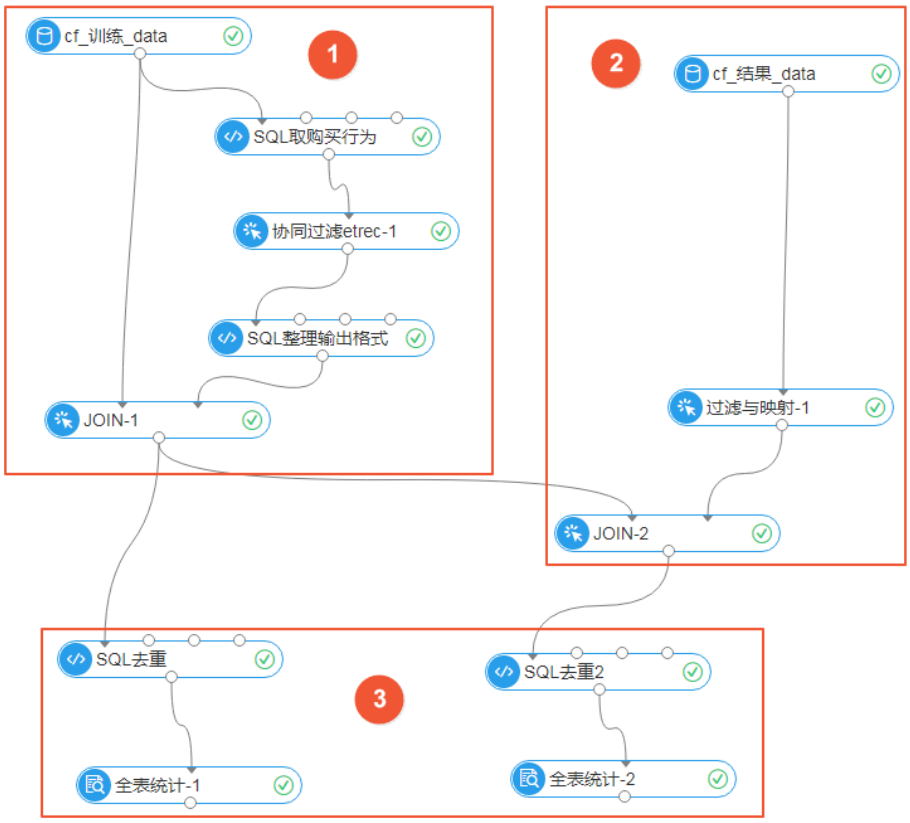
user_id	用户编号	string	购物的用户ID
item_id	物品编号	string	被购买物品的编号
active_type	购物行为	string	0表示点击，1表示购买，2表示收藏，3表示购物车
active_date	购物时间	string	购物发生的时间

数据截图如下。

10944750	8689	2	5月2日
10944750	25687	2	5月8日
10944750	7150	1	6月7日
10944750	13451	0	6月4日
10944750	13451	0	6月4日
10944750	13451	0	6月4日
10944750	13451	0	6月4日
10944750	13451	0	6月4日

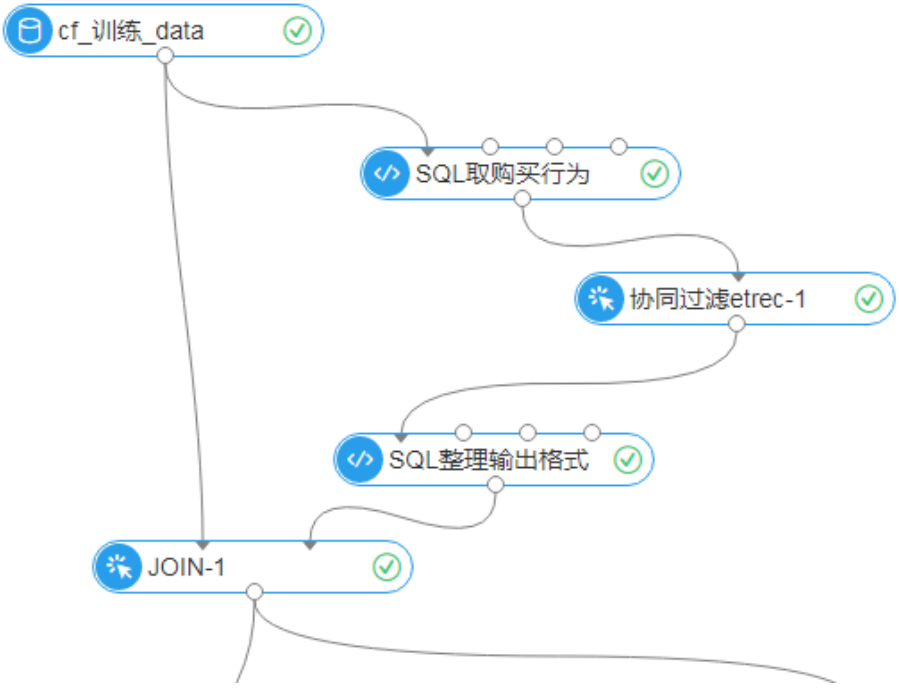
数据探索流程

实验流程图如下。



- 1：根据关联规则生成推荐列表
- 2：七月之后的真实购物行为
- 3：推荐数和命中数的统计

1. 生成推荐列表



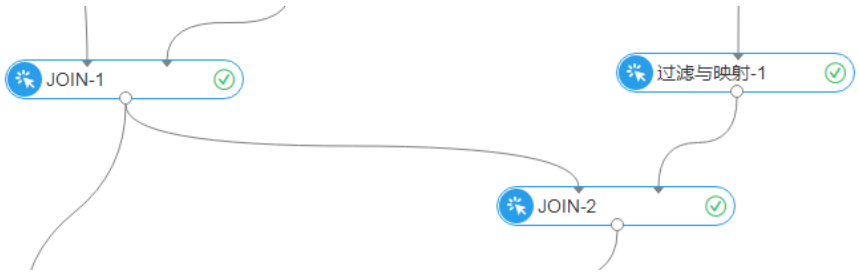
输入的数据源是7月份之前的购物行为数据，通过SQL脚本取出用户的购买行为数据，进入协同过滤组件。协同过滤组件设置中把TopN设置成1，表示每个item返回最相近的item和它的权重。通过购买行为，分析出哪些商品被同一个user购买的可能性最大，如下图所示。

协同过滤结果表示的是商品的关联性，“itemid”表示目标商品，“similarity”字段冒号的左侧表示与目标关联性高的商品，右侧表示两个商品的关联性概率。

itemid ▲	similarity ▲
1000	15584:0.2747133918
10014	18712:0.05229603127
10066	3228:0.2650900672
1008	24507:1
10082	18024:0.1781525919
1010	18024:0.2104947227
10133	14020:0.2070609237
1015	18024:0.2104947227
10151	26288:0.4366713611
10171	11080:0.2401992435

2. 推荐

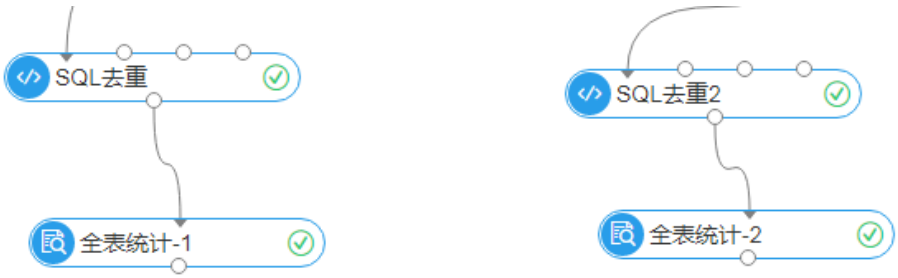
步骤一介绍了如何生成强关联商品的对应列表。这里使用了比较简单的推荐规则，比如用户甲在7月份之前买了商品A，商品A与B强相关，我们就在7月份之后推荐了商品B，并探查这次推荐是否命中，实验流程如下图所示。



3. 结果统计

下图是统计模块，左边的全表统计组件展示的是根据7月份之前的购物行为生成的推荐列表，去重后共18065条

。右边的全表统计组件显示一共命中了90条。



推荐系统反思

根据上文的统计结果可以看出，本次试验的推荐效果并不理想，原因如下。

- 本文档只是针对了业务场景大致介绍了协同过滤推荐的用法。很多针对于购物行为推荐的关键点都没有处理，比如说时间序列。购物行为一定要注意时效性的分析，跨度达到几个月的推荐不会有好的效果。
- 本文档只考虑了商品的关联性，没有考虑推荐商品的属性，例如是高频还是低频商品。比如用户A上个月买了个手机，那下个月就不大会继续购买手机，因为手机是低频消费品。
- 基于关联规则的推荐方法最好是作为补充，真正想提高准确率还是要依靠机器学习算法训练模型的方式。

其它

请进入阿里云数加机器学习平台体验阿里云机器学习产品，并通过云栖社区公众号参与讨论。

实时热点新闻挖掘案例

PAI OnlineLearning挖掘实时热点新闻

打开新闻客户端，往往会收到热点新闻推送相关的内容。新闻客户端作为一个承载新闻的平台，实时会产生大量的

新闻，如何快速挖掘出哪些新产生的新闻会成为成为热点新闻，决定着整个平台的新闻推荐质量。

如何从平台中海量的新闻素材中找到最有潜力成为热点的新闻需要使用机器学习相关的算法，传统做法是将每天获取的历史咨询下载并且离线训练模型，再将生成的热点发现模型推上线供第二日使用。但是这种离线训练所生成的模型往往缺乏时效性的属性，因为每天热点新闻都是实时产生的，用过去的模型预测实时产生的数据显然是缺乏对数据时效性的理解。

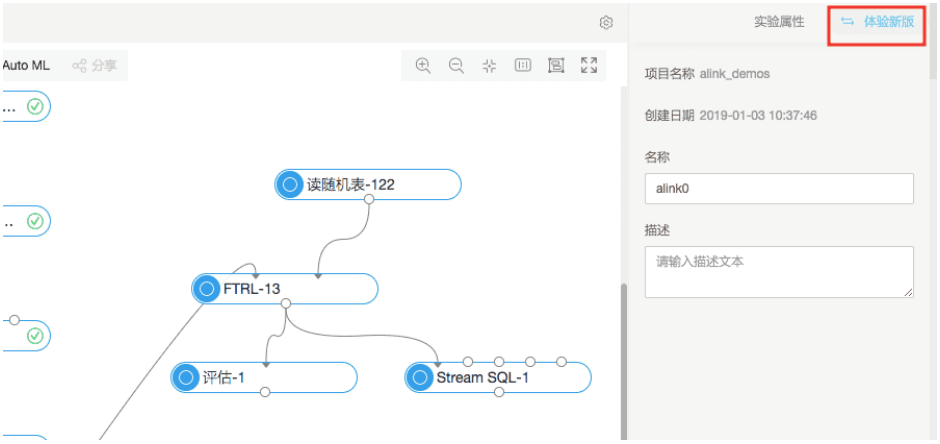
针对这种场景，PAI平台开创性的提出来Online-Learning的解决方案，通过流式算法和离线算法的结合，既能够发挥离线训练对大规模数据的强大处理能力，又能够发挥流式机器学习算法对实时模型的更新能力，做到流批同跑，完美解决模型时效性的问题。今天就以实时热点新闻挖掘案例为例，为大家介绍PAI OnlineLearning的解决方案。

实验流程

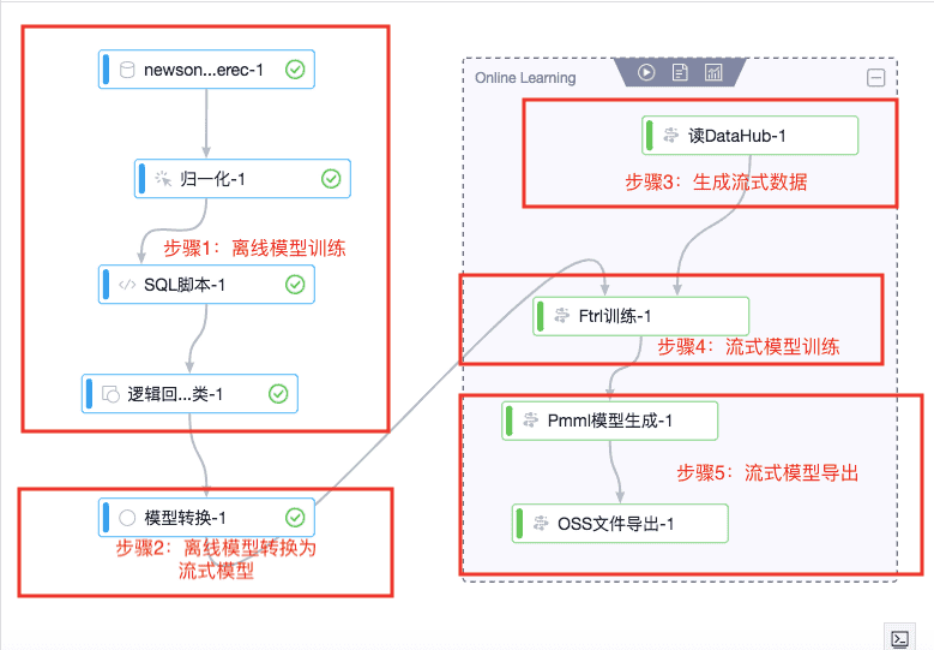
1.开通并使用

目前PAI-OnlineLearning解决方案仍处于邀请公测阶段，有需求的客户请填写问卷：

开通服务后，点击“体验新版”按钮即可开启试用



2.实验流程介绍



（注：PAI中离线计算组件用蓝色标识，流式计算组件由绿色标识，流式组件相连将形成计算组，因为流式组件需要多个组件的运行停止状态一致）

步骤1：离线模型训练

本文使用的数据是3万条来自UCI开放数据集提供的新闻文本数据。

地址：https://archive.ics.uci.edu/ml/datasets/Online+News+Popularity

数据组成：包含新闻的URL以及产生时间，另外还包含了58个特征以及1个目标值，目标值“share”是新闻的分享次数，建模过程中将share字段利用sql组件处理成一个二分类问题，新闻share次数超过10000次为热点新闻，小于10000次为非热门新闻

特征的组成如下图所示：

Feature	Type (#)
Words	
Number of words in the title	number (1)
Number of words in the article	number (1)
Average word length	number (1)
Rate of non-stop words	ratio (1)
Rate of unique words	ratio (1)
Rate of unique non-stop words	ratio (1)
Links	
Number of links	number (1)
Number of Mashable article links	number (1)
Minimum, average and maximum number of shares of Mashable links	number (3)
Digital Media	
Number of images	number (1)
Number of videos	number (1)
Time	
Day of the week	nominal (1)
Published on a weekend?	bool (1)

Feature	Type (#)
Keywords	
Number of keywords	number (1)
Worst keyword (min./avg./max. shares)	number (3)
Average keyword (min./avg./max. shares)	number (3)
Best keyword (min./avg./max. shares)	number (3)
Article category (Mashable data channel)	nominal (1)
Natural Language Processing	
Closeness to top 5 LDA topics	ratio (5)
Title subjectivity	ratio (1)
Article text subjectivity score and its absolute difference to 0.5	ratio (2)
Title sentiment polarity	ratio (1)
Rate of positive and negative words	ratio (2)
Pos. words rate among non-neutral words	ratio (1)
Neg. words rate among non-neutral words	ratio (1)
Polarity of positive words (min./avg./max.)	ratio (3)
Polarity of negative words (min./avg./max.)	ratio (3)
Article text polarity score and its absolute difference to 0.5	ratio (2)

Target	Type (#)
Number of article Mashable shares	number (1)

利用逻辑回归模型训练生成一个二分类模型，这个模型用来评估新闻是否会成为热点新闻。

（注：目前PAI OnlineLearning只支持逻辑回归算法）

步骤2：离线模型转换成流式模型

通过“模型转换”组件，可以将离线生成的逻辑回归模型转换成流式算法可读取了流式模型。

步骤3：生成流式数据

从步骤3开始就进入了流式算法组件的步骤，PAI平台提供多种流式数据源，本案例以Datahub为例。

Datahub地址：https://datahub.console.aliyun.com/datahub

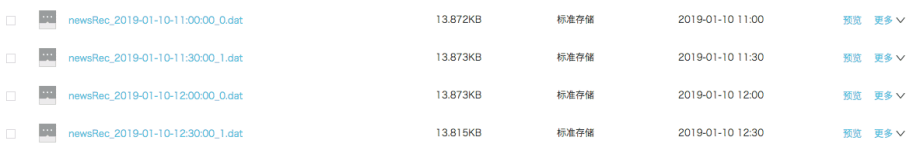
Datahub是一种流式数据对列，支持JAVA、PYTHON等多种语言采集方式，在具体使用过程中可以通过Datahub链接用户实时产生的数据以及PAI的训练服务。注意：Datahub输入的数据流格式需要与离线训练的数据流的字段完全一致，这样才可以对离线的模型进行实时更新。

步骤4：流式模型训练

FTRL算法基本等同于流式的逻辑回归算法，在使用过程中需要按照LR算法配置参数，需要注意”模型保存时间间隔参数“的配置，这个参数决定了实时计算产生模型的时间周期。

步骤5：流式模型导出

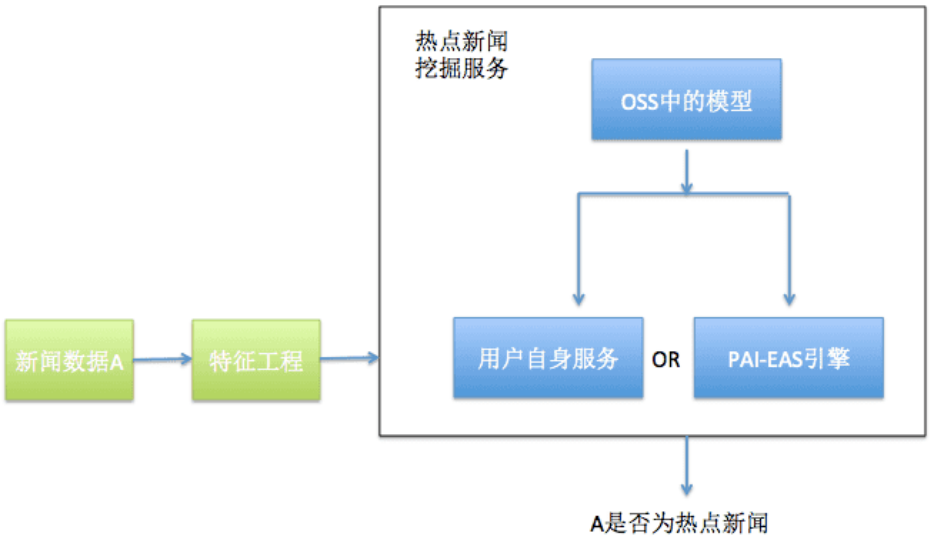
先将分类模型导出为PMML格式，然后可以支持将模型写入OSS，写入周期同模型生成的周期。写入模型示例：



如果有流式评估数据，系统也可以将实时的模型评估指标与模型一同存入OSS。

3.模型使用介绍

通过以上步骤已经产生了新闻热点预测模型，生成的模型已经存入OSS，可以直接在PAI-EAS在线预测服务引擎进行部署也可以下载下来在本地预测引擎使用。新闻数据进来后先要做特征工程（同”步骤1：离线模型训练“中的特征处理方式），然后将特征工程处理结果输入”热点新闻挖掘服务“，将会返回新闻是否是热点新闻。



图像视频分析

图像特征提取

PAI提供强大的图像特征抽取的能力，支持多机分布式运行。利用PAI，你可以方便的从图片中抽取特征，支持从oss读取图片，结果写回oss文件， 同样支持读表获取图片、结果写回表中

下面我们以oss IO通路为例介绍一下图片特征抽取的过程。

数据说明

特征提取之前，你需要把你所有的图片文件上传到oss， 然后准备一个文件列表，每一行是一个图片对应的oss地址， 格式示例如下，对应PAI命令的Dinput_oss_file文件：

```
oss://bucketname/path/to/your/img1.jpg
oss://bucketname/path/to/your/img2.jpg
oss://bucketname/path/to/your/img3.jpg
```

特征抽取PAI命令

基于上面产生的文件列表，你可以执行pai命令来进行特征抽取

```
pai -name ev_predict_ext
-Dmodel_path='oss://pai-vision-data-sh/pretrained_models/saved_models/resnet_v1_50/'
-Dmodel_type='feature_extractor'
-Dinput_oss_file='oss://{oss_bucket}/path/to/your/filelist.txt'
-Doutput_oss_file='oss://{oss_bucket}/path/to/your/result.txt'
-Dimage_type='url'
-Dfeature_name='resnet_v1_50/block4'
-Dnum_worker=2
-DcpuRequired=800
-DgpuRequired=100
-Dbuckets='oss://{oss_bucket}/'
-Darn='your_role_arn'
-DossHost='oss-cn-shanghai-internal.aliyuncs.com'
```

- -Dmodel_path：模型地址
- -Dmodel_type：图像特征抽取时固定使用’ feature_extractor’
- -Dinput_oss_file 输入文件列表oss地址
- -Doutput_oss_file：输出结果文件oss地址
- -Dimage_type：输入文件列表中图片类型，url或者base64， 支持每行为图片url，也支持每行数据为图片base64数据
- -Dfeature_name：使用模型中的哪一层输出作为特征
- -Dnum_worker：特征抽取使用的worker数目
- -DcpuRequired：每个worker上申请使用的cpu， 100表示一个cpu
- -DgpuRequired： 每个worker上申请使用的gpu， 100表示一个gpu
- -Dbuckets：根目录
- -Darn：见上文

输出结果

结果文件中每一行是一个图片的特征结果， 由文件路径和json字符串构成，示例如下：

```
oss://path/to/your/image1.jpg,  {"feature": [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.4583122730255127, 0.0]}
oss://path/to/your/image1.jpg, {"feature": [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.4583122730255127, 0.0]}
oss://path/to/your/image1.jpg, {"feature": [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.4583122730255127, 0.0]}
```

其中json字符串中只有一个k-v对，feature对应的value表示图像特征，由一个list构成，是一个一维向量

模型地址和模型输出说明

resnet_v1_50地址： oss://pai-vision-data-sh/pretrained_models/saved_models/resnet_v1_50

输出说明

```
resnet_v1_50/block1 shape: [None, 56, 56, 256] type: <dtype: 'float32'>
resnet_v1_50/block2 shape: [None, 28, 28, 512] type: <dtype: 'float32'>
resnet_v1_50/block3 shape: [None, 14, 14, 1024] type: <dtype: 'float32'>
resnet_v1_50/block4 shape: [None, 7, 7, 2048] type: <dtype: 'float32'>
AvgPool_1a shape: [None, 1, 1, 2048] type: <dtype: 'float32'>
resnet_v1_50/logits shape: [None, 1, 1, 1000] type: <dtype: 'float32'>
predictions shape: [None] type: <dtype: 'int32'>
class shape: [None] type: <dtype: 'int32'>
preprocessed_images shape: [None, 224, 224, 3] type: <dtype: 'float32'>
resnet_v1_50/conv1 shape: [None, 112, 112, 64] type: <dtype: 'float32'>
logits shape: [None, 1000] type: <dtype: 'float32'>
probs shape: [None, 1001] type: <dtype: 'float32'>
resnet_v1_50/spatial_squeeze shape: [None, 1000] type: <dtype: 'float32'>
```

resnet_v1_101地址： oss://pai-vision-data-sh/pretrained_models/saved_models/resnet_v1_101输出说明

```
resnet_v1_101/block4 shape: [None, 7, 7, 2048] type: <dtype: 'float32'>
resnet_v1_101/logits shape: [None, 1, 1, 1000] type: <dtype: 'float32'>
resnet_v1_101/block2 shape: [None, 28, 28, 512] type: <dtype: 'float32'>
resnet_v1_101/conv1 shape: [None, 112, 112, 64] type: <dtype: 'float32'>
resnet_v1_101/block1 shape: [None, 56, 56, 256] type: <dtype: 'float32'>
class shape: [None] type: <dtype: 'int32'>
resnet_v1_101/spatial_squeeze shape: [None, 1000] type: <dtype: 'float32'>
predictions shape: [None] type: <dtype: 'int32'>
preprocessed_images shape: [None, 224, 224, 3] type: <dtype: 'float32'>
```

```
logits shape: [None, 1000] type: <dtype: 'float32'>
resnet_v1_101/block3 shape: [None, 14, 14, 1024] type: <dtype: 'float32'>
probs shape: [None, 1001] type: <dtype: 'float32'>
AvgPool_1a shape: [None, 1, 1, 2048] type: <dtype: 'float32'>
```

inception_v3地址： oss://pai-vision-data-sh/pretrained_models/saved_models/inception_v3输出说明

```
preprocessed_images shape: [None, 299, 299, 3] type: <dtype: 'float32'>
Conv2d_1a_3x3 shape: [None, 149, 149, 32] type: <dtype: 'float32'>
Conv2d_2a_3x3 shape: [None, 147, 147, 32] type: <dtype: 'float32'>
Conv2d_2b_3x3 shape: [None, 147, 147, 64] type: <dtype: 'float32'>
MaxPool_3a_3x3 shape: [None, 73, 73, 64] type: <dtype: 'float32'>
Conv2d_3b_1x1 shape: [None, 73, 73, 80] type: <dtype: 'float32'>
Conv2d_4a_3x3 shape: [None, 71, 71, 192] type: <dtype: 'float32'>
MaxPool_5a_3x3 shape: [None, 35, 35, 192] type: <dtype: 'float32'>
Mixed_5b shape: [None, 35, 35, 256] type: <dtype: 'float32'>
Mixed_5c shape: [None, 35, 35, 288] type: <dtype: 'float32'>
Mixed_5d shape: [None, 35, 35, 288] type: <dtype: 'float32'>
Mixed_6a shape: [None, 17, 17, 768] type: <dtype: 'float32'>
Mixed_6b shape: [None, 17, 17, 768] type: <dtype: 'float32'>
Mixed_6c shape: [None, 17, 17, 768] type: <dtype: 'float32'>
Mixed_6d shape: [None, 17, 17, 768] type: <dtype: 'float32'>
Mixed_6e shape: [None, 17, 17, 768] type: <dtype: 'float32'>
Mixed_7a shape: [None, 8, 8, 1280] type: <dtype: 'float32'>
Mixed_7b shape: [None, 8, 8, 2048] type: <dtype: 'float32'>
Mixed_7c shape: [None, 8, 8, 2048] type: <dtype: 'float32'>
AvgPool_1a shape: [None, 1, 1, 2048] type: <dtype: 'float32'>
PreLogits shape: [None, 1, 1, 2048] type: <dtype: 'float32'>
Logits shape: [None, 1001] type: <dtype: 'float32'>
Predictions shape: [None, 1001] type: <dtype: 'float32'>
```

```
logits shape: [None, 1001] type: <dtype: 'float32'>
probs shape: [None, 1001] type: <dtype: 'float32'>
class shape: [None] type: <dtype: 'int32'>
predictions shape: [None] type: <dtype: 'int32'>
original_image shape: [None, None, None, 3] type: <dtype: 'float32'>
original_image_shape shape: [None, 3] type: <dtype: 'int32'>
```

inception_v4地址： oss://pai-vision-data-sh/pretrained_models/saved_models/inception_v4输出说明

```
preprocessed_images shape: [None, 299, 299, 3] type: <dtype: 'float32'>
Conv2d_1a_3x3 shape: [None, 149, 149, 32] type: <dtype: 'float32'>
Conv2d_2a_3x3 shape: [None, 147, 147, 32] type: <dtype: 'float32'>
Conv2d_2b_3x3 shape: [None, 147, 147, 64] type: <dtype: 'float32'>
Mixed_3a shape: [None, 73, 73, 160] type: <dtype: 'float32'>
Mixed_4a shape: [None, 71, 71, 192] type: <dtype: 'float32'>
Mixed_5a shape: [None, 35, 35, 384] type: <dtype: 'float32'>
Mixed_5b shape: [None, 35, 35, 384] type: <dtype: 'float32'>
Mixed_5c shape: [None, 35, 35, 384] type: <dtype: 'float32'>
Mixed_5d shape: [None, 35, 35, 384] type: <dtype: 'float32'>
Mixed_5e shape: [None, 35, 35, 384] type: <dtype: 'float32'>
Mixed_6a shape: [None, 17, 17, 1024] type: <dtype: 'float32'>
Mixed_6b shape: [None, 17, 17, 1024] type: <dtype: 'float32'>
Mixed_6c shape: [None, 17, 17, 1024] type: <dtype: 'float32'>
Mixed_6d shape: [None, 17, 17, 1024] type: <dtype: 'float32'>
Mixed_6e shape: [None, 17, 17, 1024] type: <dtype: 'float32'>
Mixed_6f shape: [None, 17, 17, 1024] type: <dtype: 'float32'>
Mixed_6g shape: [None, 17, 17, 1024] type: <dtype: 'float32'>
Mixed_6h shape: [None, 17, 17, 1024] type: <dtype: 'float32'>
Mixed_7a shape: [None, 8, 8, 1536] type: <dtype: 'float32'>
```

```
Mixed_7b shape: [None, 8, 8, 1536] type: <dtype: 'float32'>
Mixed_7c shape: [None, 8, 8, 1536] type: <dtype: 'float32'>
Mixed_7d shape: [None, 8, 8, 1536] type: <dtype: 'float32'>
AvgPool_1a shape: [None, 1, 1, 1536] type: <dtype: 'float32'>
PreLogitsFlatten shape: [None, 1536] type: <dtype: 'float32'>
Logits shape: [None, 1001] type: <dtype: 'float32'>
Predictions shape: [None, 1001] type: <dtype: 'float32'>
logits shape: [None, 1001] type: <dtype: 'float32'>
probs shape: [None, 1001] type: <dtype: 'float32'>
class shape: [None] type: <dtype: 'int32'>
predictions shape: [None] type: <dtype: 'int32'>
original_image shape: [None, None, None, 3] type: <dtype: 'float32'>
original_image_shape shape: [None, 3] type: <dtype: 'int32'>
```

mobilenet_v2地址： oss://pai-vision-data-sh/pretrained_models/saved_models/mobilenet_v2_1.0_224输出说明

```
preprocessed_images shape: [None, 224, 224, 3] type: <dtype: 'float32'>
layer_1 shape: [None, 112, 112, 32] type: <dtype: 'float32'>
layer_2 shape: [None, 112, 112, 16] type: <dtype: 'float32'>
layer_3 shape: [None, 56, 56, 24] type: <dtype: 'float32'>
layer_4 shape: [None, 56, 56, 24] type: <dtype: 'float32'>
layer_5 shape: [None, 28, 28, 32] type: <dtype: 'float32'>
layer_6 shape: [None, 28, 28, 32] type: <dtype: 'float32'>
layer_7 shape: [None, 28, 28, 32] type: <dtype: 'float32'>
layer_8 shape: [None, 14, 14, 64] type: <dtype: 'float32'>
layer_9 shape: [None, 14, 14, 64] type: <dtype: 'float32'>
layer_10 shape: [None, 14, 14, 64] type: <dtype: 'float32'>
layer_11 shape: [None, 14, 14, 64] type: <dtype: 'float32'>
layer_12 shape: [None, 14, 14, 96] type: <dtype: 'float32'>
```

layer_13 shape: [None, 14, 14, 96] type: <dtype: 'float32'>

layer_14 shape: [None, 14, 14, 96] type: <dtype: 'float32'>

layer_15 shape: [None, 7, 7, 160] type: <dtype: 'float32'>

layer_16 shape: [None, 7, 7, 160] type: <dtype: 'float32'>

layer_17 shape: [None, 7, 7, 160] type: <dtype: 'float32'>

layer_18 shape: [None, 7, 7, 320] type: <dtype: 'float32'>

layer_19 shape: [None, 7, 7, 1280] type: <dtype: 'float32'>

layer_2/depthwise_output shape: [None, 112, 112, 32] type: <dtype: 'float32'>

layer_2/output shape: [None, 112, 112, 16] type: <dtype: 'float32'>

layer_3/expansion_output shape: [None, 112, 112, 96] type: <dtype: 'float32'>

layer_3/depthwise_output shape: [None, 56, 56, 96] type: <dtype: 'float32'>

layer_3/output shape: [None, 56, 56, 24] type: <dtype: 'float32'>

layer_4/expansion_output shape: [None, 56, 56, 144] type: <dtype: 'float32'>

layer_4/depthwise_output shape: [None, 56, 56, 144] type: <dtype: 'float32'>

layer_4/output shape: [None, 56, 56, 24] type: <dtype: 'float32'>

layer_5/expansion_output shape: [None, 56, 56, 144] type: <dtype: 'float32'>

layer_5/depthwise_output shape: [None, 28, 28, 144] type: <dtype: 'float32'>

layer_5/output shape: [None, 28, 28, 32] type: <dtype: 'float32'>

layer_6/expansion_output shape: [None, 28, 28, 192] type: <dtype: 'float32'>

layer_6/depthwise_output shape: [None, 28, 28, 192] type: <dtype: 'float32'>

layer_6/output shape: [None, 28, 28, 32] type: <dtype: 'float32'>

layer_7/expansion_output shape: [None, 28, 28, 192] type: <dtype: 'float32'>

layer_7/depthwise_output shape: [None, 28, 28, 192] type: <dtype: 'float32'>

layer_7/output shape: [None, 28, 28, 32] type: <dtype: 'float32'>

layer_8/expansion_output shape: [None, 28, 28, 192] type: <dtype: 'float32'>

layer_8/depthwise_output shape: [None, 14, 14, 192] type: <dtype: 'float32'>

layer_8/output shape: [None, 14, 14, 64] type: <dtype: 'float32'>

layer_9/expansion_output shape: [None, 14, 14, 384] type: <dtype: 'float32'>

```
layer_9/depthwise_output shape: [None, 14, 14, 384] type: <dtype: 'float32'>
layer_9/output shape: [None, 14, 14, 64] type: <dtype: 'float32'>
layer_10/expansion_output shape: [None, 14, 14, 384] type: <dtype: 'float32'>
layer_10/depthwise_output shape: [None, 14, 14, 384] type: <dtype: 'float32'>
layer_10/output shape: [None, 14, 14, 64] type: <dtype: 'float32'>
layer_11/expansion_output shape: [None, 14, 14, 384] type: <dtype: 'float32'>
layer_11/depthwise_output shape: [None, 14, 14, 384] type: <dtype: 'float32'>
layer_11/output shape: [None, 14, 14, 64] type: <dtype: 'float32'>
layer_12/expansion_output shape: [None, 14, 14, 384] type: <dtype: 'float32'>
layer_12/depthwise_output shape: [None, 14, 14, 384] type: <dtype: 'float32'>
layer_12/output shape: [None, 14, 14, 96] type: <dtype: 'float32'>
layer_13/expansion_output shape: [None, 14, 14, 576] type: <dtype: 'float32'>
layer_13/depthwise_output shape: [None, 14, 14, 576] type: <dtype: 'float32'>
layer_13/output shape: [None, 14, 14, 96] type: <dtype: 'float32'>
layer_14/expansion_output shape: [None, 14, 14, 576] type: <dtype: 'float32'>
layer_14/depthwise_output shape: [None, 14, 14, 576] type: <dtype: 'float32'>
layer_14/output shape: [None, 14, 14, 96] type: <dtype: 'float32'>
layer_15/expansion_output shape: [None, 14, 14, 576] type: <dtype: 'float32'>
layer_15/depthwise_output shape: [None, 7, 7, 576] type: <dtype: 'float32'>
layer_15/output shape: [None, 7, 7, 160] type: <dtype: 'float32'>
layer_16/expansion_output shape: [None, 7, 7, 960] type: <dtype: 'float32'>
layer_16/depthwise_output shape: [None, 7, 7, 960] type: <dtype: 'float32'>
layer_16/output shape: [None, 7, 7, 160] type: <dtype: 'float32'>
layer_17/expansion_output shape: [None, 7, 7, 960] type: <dtype: 'float32'>
layer_17/depthwise_output shape: [None, 7, 7, 960] type: <dtype: 'float32'>
layer_17/output shape: [None, 7, 7, 160] type: <dtype: 'float32'>
layer_18/expansion_output shape: [None, 7, 7, 960] type: <dtype: 'float32'>
layer_18/depthwise_output shape: [None, 7, 7, 960] type: <dtype: 'float32'>
```

```
layer_18/output shape: [None, 7, 7, 320] type: <dtype: 'float32'>
AvgPool_1a shape: [None, 1, 1, 1280] type: <dtype: 'float32'>
Logits shape: [None, 1001] type: <dtype: 'float32'>
Predictions shape: [None, 1001] type: <dtype: 'float32'>
logits shape: [None, 1001] type: <dtype: 'float32'>
probs shape: [None, 1001] type: <dtype: 'float32'>
class shape: [None] type: <dtype: 'int32'>
predictions shape: [None] type: <dtype: 'int32'>
original_image shape: [None, None, None, 3] type: <dtype: 'float32'>
original_image_shape shape: [None, 3] type: <dtype: 'int32'>
```

efficientnet_b0地址： oss://pai-vision-data-sh/pretrained_models/saved_models/efficientnet-b0输出说明

```
stem shape: [None, 112, 112, 32] type: <dtype: 'float32'>
block_0/expansion_output shape: [None, 112, 112, 32] type: <dtype: 'float32'>
block_0 shape: [None, 112, 112, 16] type: <dtype: 'float32'>
reduction_1/expansion_output shape: [None, 112, 112, 32] type: <dtype: 'float32'>
reduction_1 shape: [None, 112, 112, 16] type: <dtype: 'float32'>
block_1/expansion_output shape: [None, 56, 56, 96] type: <dtype: 'float32'>
block_1 shape: [None, 56, 56, 24] type: <dtype: 'float32'>
block_2/expansion_output shape: [None, 56, 56, 144] type: <dtype: 'float32'>
block_2 shape: [None, 56, 56, 24] type: <dtype: 'float32'>
reduction_2/expansion_output shape: [None, 56, 56, 144] type: <dtype: 'float32'>
reduction_2 shape: [None, 56, 56, 24] type: <dtype: 'float32'>
block_3/expansion_output shape: [None, 28, 28, 144] type: <dtype: 'float32'>
block_3 shape: [None, 28, 28, 40] type: <dtype: 'float32'>
block_4/expansion_output shape: [None, 28, 28, 240] type: <dtype: 'float32'>
block_4 shape: [None, 28, 28, 40] type: <dtype: 'float32'>
reduction_3/expansion_output shape: [None, 28, 28, 240] type: <dtype: 'float32'>
```

```
reduction_3 shape: [None, 28, 28, 40] type: <dtype: 'float32'>
block_5/expansion_output shape: [None, 14, 14, 240] type: <dtype: 'float32'>
block_5 shape: [None, 14, 14, 80] type: <dtype: 'float32'>
block_6/expansion_output shape: [None, 14, 14, 480] type: <dtype: 'float32'>
block_6 shape: [None, 14, 14, 80] type: <dtype: 'float32'>
block_7/expansion_output shape: [None, 14, 14, 480] type: <dtype: 'float32'>
block_7 shape: [None, 14, 14, 80] type: <dtype: 'float32'>
block_8/expansion_output shape: [None, 14, 14, 480] type: <dtype: 'float32'>
block_8 shape: [None, 14, 14, 112] type: <dtype: 'float32'>
block_9/expansion_output shape: [None, 14, 14, 672] type: <dtype: 'float32'>
block_9 shape: [None, 14, 14, 112] type: <dtype: 'float32'>
block_10/expansion_output shape: [None, 14, 14, 672] type: <dtype: 'float32'>
block_10 shape: [None, 14, 14, 112] type: <dtype: 'float32'>
reduction_4/expansion_output shape: [None, 14, 14, 672] type: <dtype: 'float32'>
reduction_4 shape: [None, 14, 14, 112] type: <dtype: 'float32'>
block_11/expansion_output shape: [None, 7, 7, 672] type: <dtype: 'float32'>
block_11 shape: [None, 7, 7, 192] type: <dtype: 'float32'>
block_12/expansion_output shape: [None, 7, 7, 1152] type: <dtype: 'float32'>
block_12 shape: [None, 7, 7, 192] type: <dtype: 'float32'>
block_13/expansion_output shape: [None, 7, 7, 1152] type: <dtype: 'float32'>
block_13 shape: [None, 7, 7, 192] type: <dtype: 'float32'>
block_14/expansion_output shape: [None, 7, 7, 1152] type: <dtype: 'float32'>
block_14 shape: [None, 7, 7, 192] type: <dtype: 'float32'>
block_15/expansion_output shape: [None, 7, 7, 1152] type: <dtype: 'float32'>
block_15 shape: [None, 7, 7, 320] type: <dtype: 'float32'>
reduction_5/expansion_output shape: [None, 7, 7, 1152] type: <dtype: 'float32'>
reduction_5 shape: [None, 7, 7, 320] type: <dtype: 'float32'>
features shape: [None, 7, 7, 320] type: <dtype: 'float32'>
```

```
head_1x1 shape: [None, 7, 7, 1280] type: <dtype: 'float32'>

pooled_features shape: [None, 1280] type: <dtype: 'float32'>

global_pool shape: [None, 1280] type: <dtype: 'float32'>

class shape: [None] type: <dtype: 'int32'>

head shape: [None, 1000] type: <dtype: 'float32'>

logits shape: [None, 1000] type: <dtype: 'float32'>

probs shape: [None, 1001] type: <dtype: 'float32'>

predictions shape: [None] type: <dtype: 'int32'>
```

视频分类

PAI平台提供视频分类相关算法，可以基于用户的短视频数据生成视频分类模型，支持千万级别差大规模的视频样本训练，生成的模型可以部署到PAI-EAS成为Restful API服务供调用。视频分类组件包含：数据格式转换和模型训练两部分。服务调用方式可使用MaxCompute console工具或者Dataworks SQL节点，详见：https://help.aliyun.com/document_detail/154185.html

实验提供一份demo数据，数据说明见下文，另外实验会需要一个与训练的模型，以及一个配置文件和一个数据地址文件。

下载地址：https://help.aliyun.com/document_detail/155513.htm

数据说明

原始数据可以是avi、mp4等常见格式的视频，本案例提供两份数据做分类模型训练，分别是eyemakeup和lipsmakeup，具体视频截图如下图所示：

数据格式转换

数据格式转换模块可以将原始的视频文件转换为tfrecord格式，这个格式的文件可以加快模型训练的速度。数据转换的命令如下：

```
pai -name easy_vision_ext
-project algo_public
-Dbuckets='oss://{bucket_name}.{oss_host}/{path}/'
-Darn='acs:ram::*******:role/aliyunodpspaidefaultrole'
-DossHost='{oss_host}'
-Dcmd convert
-Dconvert_config='{bucket_name}.{oss_host}/{path}/{config_file}'
-Dlabel_file='{bucket_name}.{oss_host}/{path}/{config_file}/{label_file}'
-Doutput_tfrecord='{bucket_name}.{oss_host}/{path}/'
```

例如：

```
pai -name easy_vision_ext
-project algo_public
-Dbuckets='oss://demo-yuze.oss-cn-beijing-internal.aliyuncs.com/vip/'
-Darn='acs:ram::******:role/aliyunodpspaidefaultrole'
-DossHost='oss-cn-beijing-internal.aliyuncs.com'
-Dcmd convert
-Dconvert_config='oss://demo-yuze.oss-cn-beijing-internal.aliyuncs.com/vip/ucf101_qince.config'
-Dlabel_file='oss://demo-yuze.oss-cn-beijing-internal.aliyuncs.com/vip/vip.csv'
-Doutput_tfrecord='oss://demo-yuze.oss-cn-beijing-internal.aliyuncs.com/vip/'
```

- Dbuckets：OSS地址的根目录
- Darn：访问OSS的授权，可以在https://help.aliyun.com/document_detail/154186.html 的IO相关参数说明中找到获取方法
- DossHost：OSS的host地址
- Dconvert_config：配置文件，本文开篇提供了下载地址，在配置文件中需要标记类别的种类，如下

```
class_map {
label_name: "ApplyEyeMakeup"
}
class_map {
label_name: "ApplyLipstick"
}


model_type: VIDEO_CLASSIFICATION
converter_class: "QinceConverter"
write_thread_num: 8
part_record_num: 64
test_ratio: 0.0
```

Dlabel_file：训练视频的所在OSS地址，需要将视频上传到OSS并在文件中注明路径，例如，需要替换成自己的oss路径：

```
数据ID,原始数据,融合答案
1,"{""tfspath"": ""oss://demo-yuze/data/eye/public_v_ApplyEyeMakeup_g01_c01.avi""}","{""option"":
""ApplyEyeMakeup""}"
2,"{""tfspath"": ""oss://demo-yuze/data/eye/public_v_ApplyEyeMakeup_g02_c03.avi""}","{""option"":
```

```
""ApplyEyeMakeup""}"
3,"{""tfspath"": ""oss://demo-yuze/data/eye/public_v_ApplyEyeMakeup_g02_c04.avi""}","{""option"":
""ApplyEyeMakeup""}"
4,"{""tfspath"": ""oss://demo-yuze/data/eye/public_v_ApplyEyeMakeup_g03_c01.avi""}","{""option"":
""ApplyEyeMakeup""}"
5,"{""tfspath"": ""oss://demo-yuze/data/eye/public_v_ApplyEyeMakeup_g04_c01.avi""}","{""option"":
""ApplyEyeMakeup""}"
6,"{""tfspath"": ""oss://demo-yuze/data/lips/public_v_ApplyLipstick_g04_c02.avi""}","{""option"":
""ApplyEyeMakeup""}"
7,"{""tfspath"": ""oss://demo-yuze/data/lips/public_v_ApplyLipstick_g05_c01.avi""}","{""option"":
""ApplyLipstick""}"
8,"{""tfspath"": ""oss://demo-yuze/data/lips/public_v_ApplyLipstick_g07_c04.avi""}","{""option"":
""ApplyLipstick""}"
9,"{""tfspath"": ""oss://demo-yuze/data/lips/public_v_ApplyLipstick_g01_c02.avi""}","{""option"":
""ApplyLipstick""}"
```

Doutput_tfrecord：tfrecord的输出路径

视频分类模型训练

基于数据转换生成的数据，训练分类模型。命令参数：

```
pai -name ev_video_classification_ext
-project algo_public
-Dbackbone='resnet_3d_50'
-Dnum_epochs=50
-Ddecay_epochs=5
-Dsave_checkpoints_epochs=1
-Dmodel_dir='{bucket_name}.{oss_host}/{output_model_path}/'
-Duse_pretrained_model=true
-Dpretrained_model='{bucket_name}.{oss_host}/{model_path}/resent_3d_50_model.ckpt'
-Dtrain_data='{bucket_name}.{oss_host}/{path}/data_train_0_0.tfrecord'
-Dtest_data='{bucket_name}.{oss_host}/{path}/data_train_0_0.tfrecord'
-Dlabel_map_path='{bucket_name}.{oss_host}/{path}/data_label_map.pbtxt'
-Dnum_test_example=10
-Dtrain_batch_size=2
-Dtest_batch_size=2
-Dbuckets='{bucket_name}.{oss_host}/{path}'
-Darn='acs:ram::*********:role/aliyunodpspaidefaultrole'
-DossHost='{oss_host}'
-Dinitial_learning_rate=0.0001
-Dstaircase=false
-DgpuRequired=100
-Dnum_classes=2
```

示例：

```
pai -name ev_video_classification_ext
-project algo_public
-Dbackbone='resnet_3d_50'
-Dnum_epochs=50
```

```
-Ddecay_epochs=5
-Dsave_checkpoints_epochs=1
-Dmodel_dir='oss://demo-yuze.oss-cn-beijing-internal.aliyuncs.com/model/'
-Duse_pretrained_model=true
-Dpretrained_model='oss://demo-yuze.oss-cn-beijing-internal.aliyuncs.com/model/resent_3d_50_model.ckpt'
-Dtrain_data='oss://demo-yuze.oss-cn-beijing-internal.aliyuncs.com/vip/data_train_0_0.tfrecord'
-Dtest_data='oss://demo-yuze.oss-cn-beijing-internal.aliyuncs.com/vip/data_train_0_0.tfrecord'
-Dlabel_map_path='oss://demo-yuze.oss-cn-beijing-internal.aliyuncs.com/vip/data_label_map.pbtxt'
-Dnum_test_example=10
-Dtrain_batch_size=2
-Dtest_batch_size=2
-Dbuckets='oss://demo-yuze.oss-cn-beijing-internal.aliyuncs.com/vip/'
-Darn='acs:ram::********:role/aliyunodpspaidefaultrole'
-DossHost='oss-cn-beijing-internal.aliyuncs.com'
-Dinitial_learning_rate=0.0001
-Dstaircase=false
-DgpuRequired=100
-Dnum_classes=2
```

- Dbackbone：选用的网络类型
- Dmodel_dir：输出的模型地址
- Dpretrained_model：上传的预训练模型地址，预训练模型上文给出了下载地址
- Dtrain_data：数据转换生成的tfrecord文件
- Dtest_data：数据转换生成的tfrecord文件
- Dlabel_map_path：数据转换生成的.pbtxt文件
- Dnum_test_example：测试的样本数
- Dtrain_batch_size：每次参与训练的样本数
- Dbuckets：根目录
- Darn：见上文
- Dnum_classes：分类个数

最终生成的模型会是Tensorflow的savemodel格式模型，可以在Dmodel_dir找到，该模型可以部署到PAI-EAS成为RestfulAPI服务，详见PAI-EAS文档：https://help.aliyun.com/document_detail/110985.html

智能风控解决方案

订单风险识别（PAI+风险识别产品）

背景概述

风控涉及到日常生活行为的各个方面，例如线上交易订单、注册账号、账号登录等行为都会涉及到风控的校验。而一个完整的风控校验除了要包含风控领域常见的模型或者数据、规则的匹配，也要符合用户自身的业务个性化特点。

举个例子，下图是一份打标好的用户订单数据，

eventcode	email	price	amount	accountId	currency	f_1	f_2	f_3	label
de_atkpxx0...	dhdienvehi@mailme.lv	1715	4	18833421	CNY	51	72	20	0
de_atkpxx0...	oejdkfnvkf@emailwar...	1034	6	17276493	CNY	78	75	20	0
de_atkpxx0...	vkjikdnf@saynotospa...	1209	2	21513600	CNY	60	85	21	0
de_atkpxx0...	ooiwiedcv@fakedemai...	1485	3	27555049	CNY	55	87	27	0
de_atkpxx0...	ldkfjfhdhdeud@short...	625	7	11195456	CNY	79	74	42	1
de_atkpxx0...	cvfdmvkfvg@gishpupp...	460	7	28132259	CNY	51	100	77	1
de_atkpxx0...	owfvkjvir@mailinator.c...	1737	10	14544911	CNY	52	75	68	1

在做订单风险识别的时候，既有基于邮箱格式的一些风控领域常规逻辑，比如会校验邮箱的格式或者命名是否是真实的、校验邮箱是否为空等。也要结合用户自己的行为逻辑去判断，比如f_1、f_2、f_3这些用户自己的业务逻辑数据如何应用到风控中，是否能基于这些数据定制化模型。

如果要完成以上的风控任务，既要兼顾风控领域的规则性，也要兼顾用户自身业务的灵活性，需要将阿里云风险识别产品和阿里云机器学习PAI联合应用。风险识别产品提供风险领域的业务判断逻辑，PAI提供基于用户自身的灵活定制化能力，本文将针对下单风险识别这个经典场景进行介绍。

阿里云风险识别产品：https://www.aliyun.com/product/saf

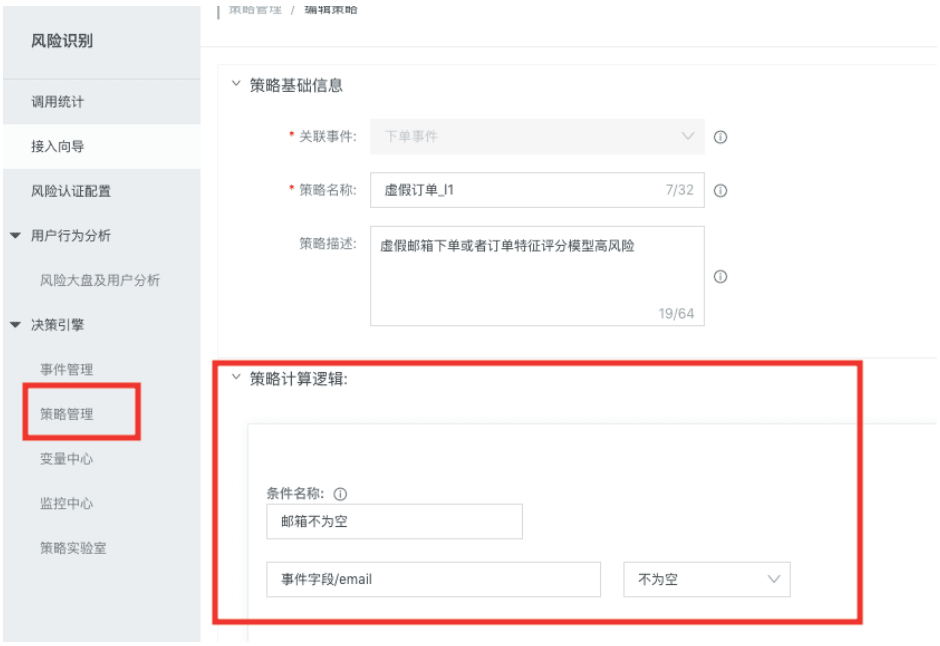
阿里云机器学习PAI：https://data.aliyun.com/product/learn

1.构建风险事件

1.进入风险识别产品，新建事件，下单行为可以看作是一种事件。

在创建事件的时间需要将数据中可能出现的字段全部添加进去，比如下单行为包含以下字段：eventCode,email,accountId,price,currency,amount,f_1,f_2,f_3，都需要添加到事件的输入字段中。

2.要在事件下新建一个策略，在策略中可以加入判断邮箱是否为空以及邮箱是否是真实邮箱等策略，这些策略都已经内置到了风险识别产品内。

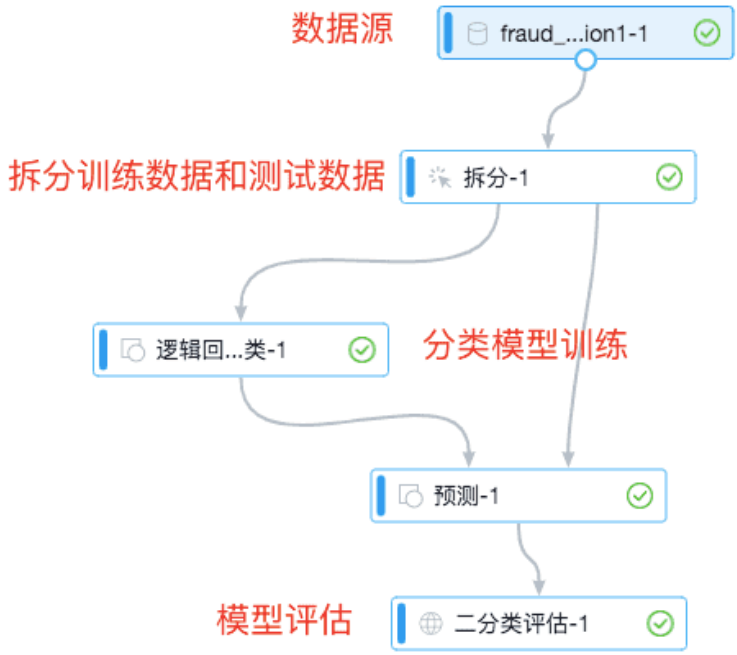


接下来要做的一件事情是进入PAI，然后结合用户的业务数据新建一个风险识别模型加入到现在的策略中。

2.用户自定义模型创建

首先创建PAI-Studio项目，并将用户的数据添加到PAI-Studio的底层计算引擎MaxCompute。

进入项目，基于PAI-Studio的机器学习组件搭建实验，并且生成模型。本次下单案例，以用户的price、amount，还有f_1、f_2、f_3等行为数据为特征，以最后的label为目标值，基于逻辑回归算法生成一个二分类模型，这个模型可以基于用户的数据去按照风险程度做打分。

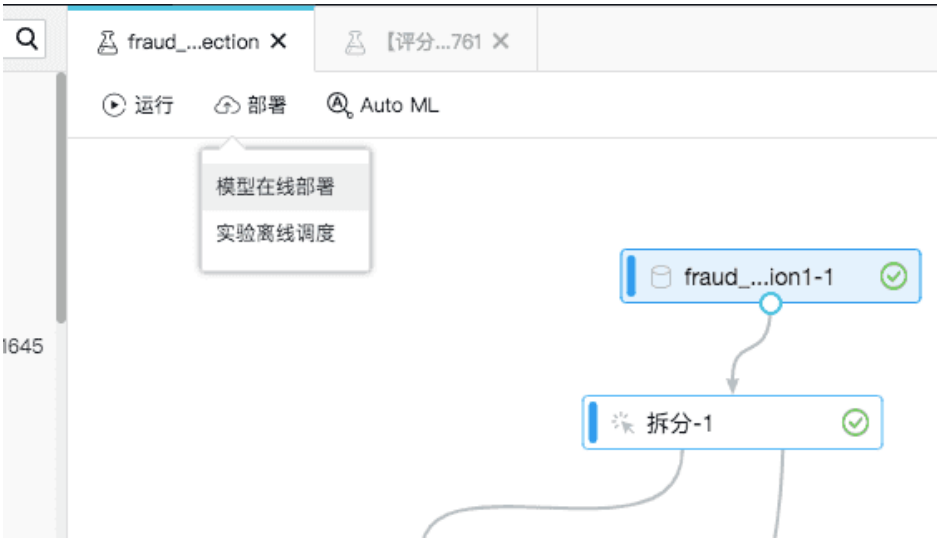


可以右键数据源查看数据：

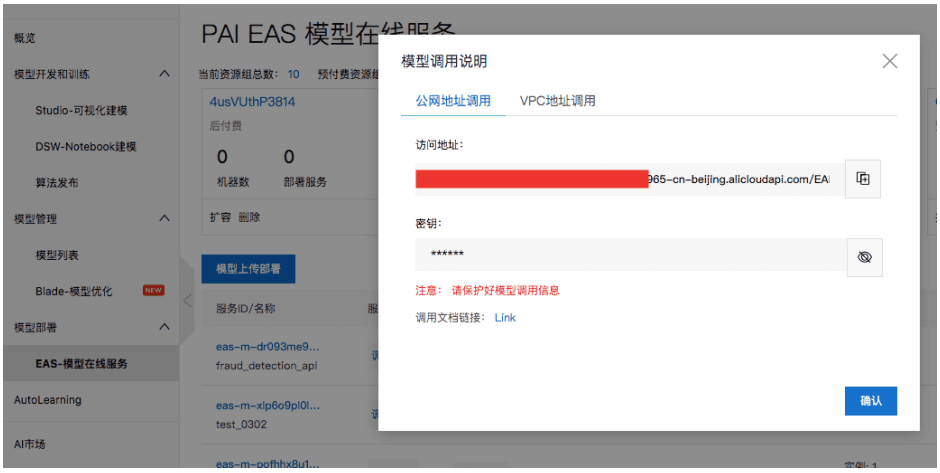
eventcode	email	price	amount	accountid	currency	f_1	f_2	f_3	label
de_atkpxx0...	dhdien...	1715	4	18833421	CNY	51	72	20	0
de_atkpxx0...	oejdkf...	1034	6	17276493	CNY	78	75	20	0
de_atkpxx0...	vkjikdn...	1209	2	21513600	CNY	60	85	21	0
de_atkpxx0...	ooiwie...	1485	3	27555049	CNY	55	87	27	0
de_atkpxx0...	ldkfjfh...	625	7	11195456	CNY	79	74	42	1
de_atkpxx0...	cvfdm...	460	7	28132259	CNY	51	100	77	1
de_atkpxx0...	owfvkj...	1737	10	14544911	CNY	52	75	68	1
de_atkpxx0...	qweor...	993	8	21515446	CNY	67	63	76	0
de_atkpxx0...	qweor...	1064	9	17048895	CNY	63	74	37	1
de_atkpxx0...	fotqfq...	1126	1	13708501	CNY	75	95	54	0
de_atkpxx0...	qhtmw...	888	3	18352454	CNY	61	76	71	0
de_atkpxx0...	juvlhg...	1002	2	29620490	CNY	64	89	78	1
de_atkpxx0...	viaqtx...	1040	9	19455552	CNY	58	88	43	0

可以在二分类评估组件处查看模型评估报告(演示数据，效果不保证)：

最终生成的评估模型需要发布到PAI-EAS成为一个Restful-API服务，在实验上方点击“模型在线部署”即可，



这时候会跳到PAI-EAS，把模型部署成一个RestfulAPI，找到模型并且在“调用信息”中点击生成公网地址，



这个时候在PAI部分的建模工作就结束了，接下来就是如何将模型服务部署到风险识别产品中。

3.网关配置

现在已经把模型变成了API服务，如果想在风险识别产品中获取模型，还需要在阿里云API网关做配置。

进入API Gateway：https://www.aliyun.com/product/apigateway

找到PAI-EAS发布过来的API，并且要做3处编辑：

1.请求模式要设置为：入参映射（过滤未知参数）

2.把模型训练涉及的参数添加进去，要选择query类型

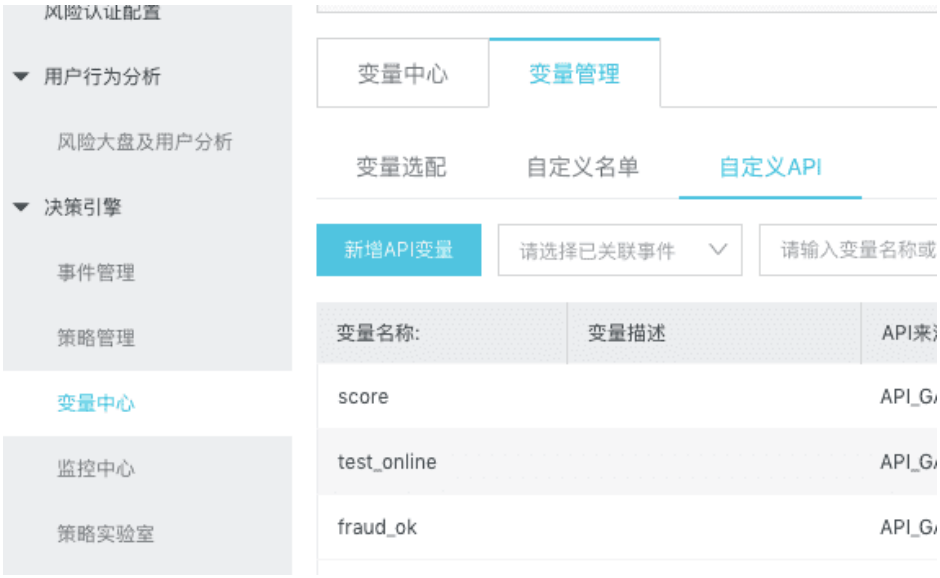
3.请求Body勾选上“非Form表单数据”

4.增加风险识别的策略变量

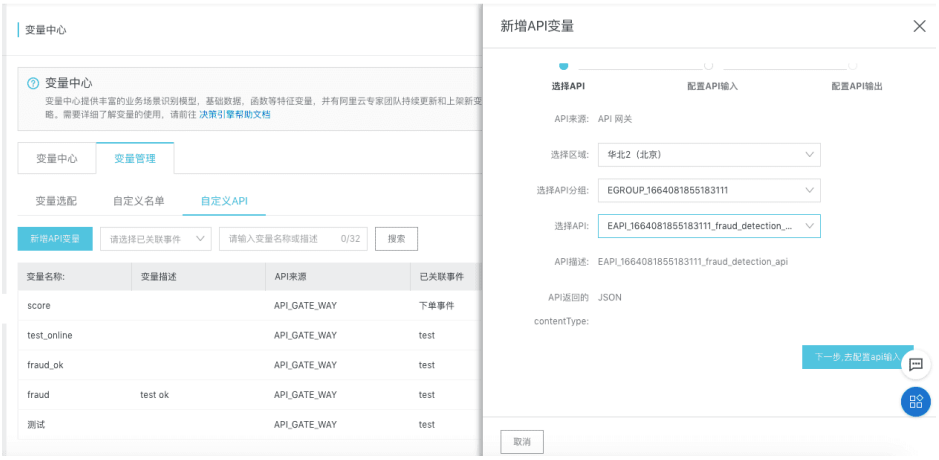
完成以上操作后，PAI生成的用户自定义模型已经可以被风险识别产品加载，

1.具体方法是进入“变量中心”选择“变量管理”。

2.选择自定义API

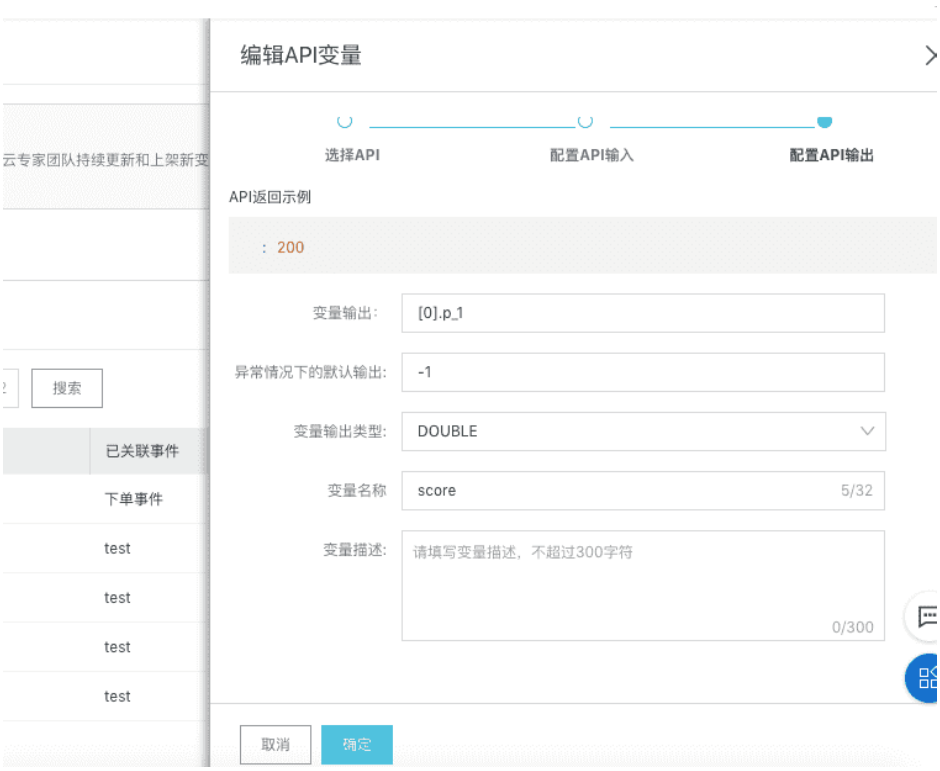


3.新建API变量并且选择API网关中配置好的服务

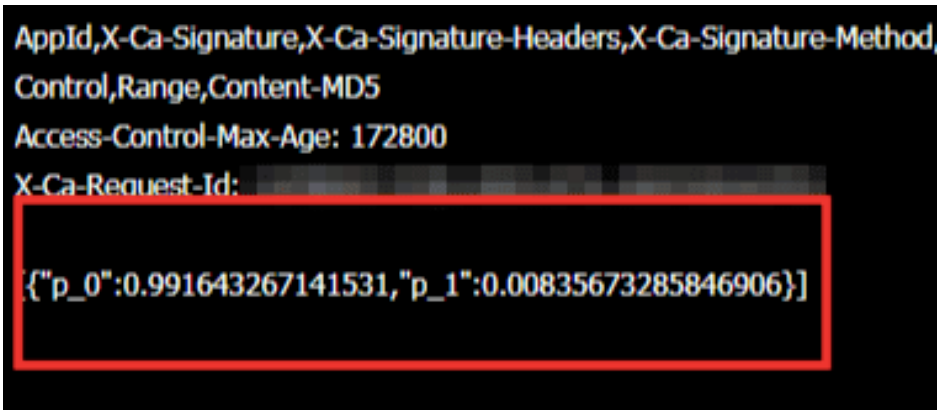


4.选择对应的事件中的每个参数和PAI中的模型的映射关系

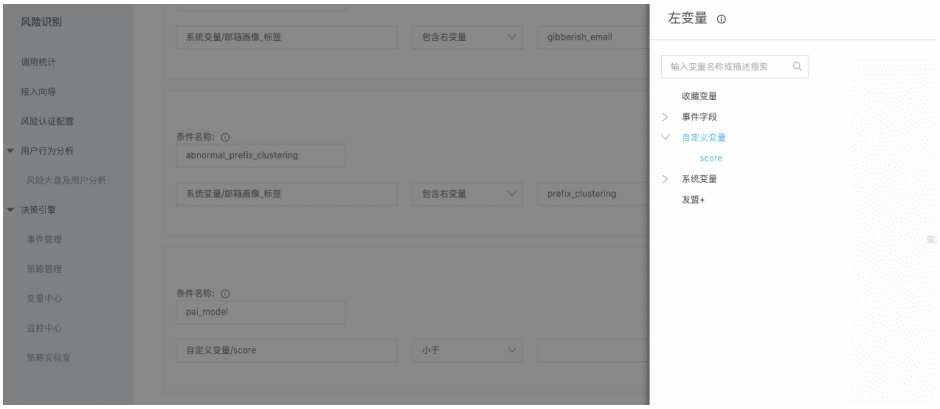
5.配置输出



需要注意的是变量输出这个字段，需要跟PAI-EAS的输出结果对应上。在二分类场景下，EAS输出的是一个数组，分别由p_0和p_1两个类别(如下图)组成，所以在这里可以通过[0].p_1或者[0].p_0指定想要的输出结果。



6.配置完API变量后，回到风险识别产品刚才设置好的“策略管理”页面中，将刚才配置好的API变量加到策略中去，需要选择自定义变量的具体返回变量名称。



7.最后要设计计算逻辑，因为风险识别产品中包含多种邮箱相关的策略，这些策略和用户基于PAI自定义生成的模型之间的执行顺序需要做一个配置。

提交即完成了整个链路的配置。

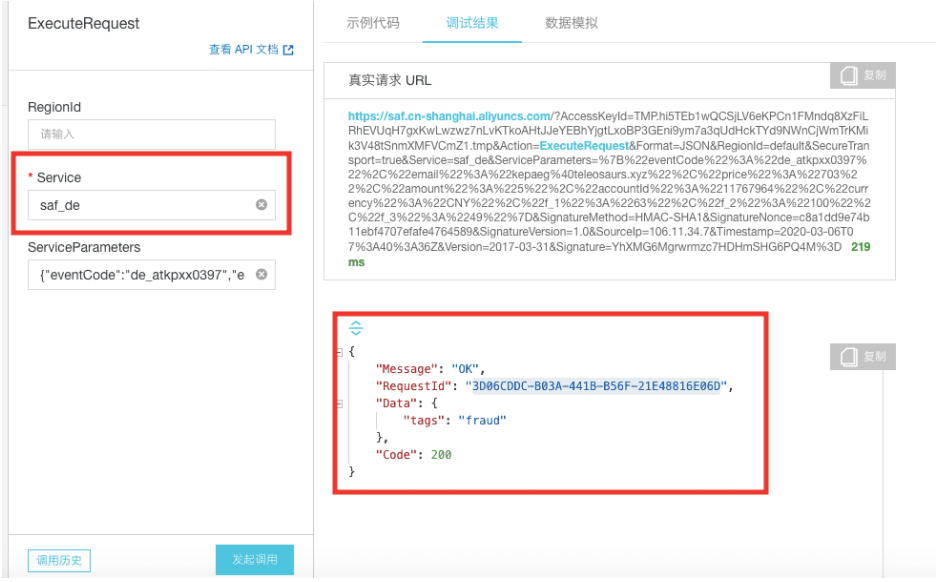
5.订单风险控制服务测试

可以进入Open API的线上地址测试整个服务的效果，地址：
https://api.aliyun.com/#/?product=saf&api=ExecuteRequest

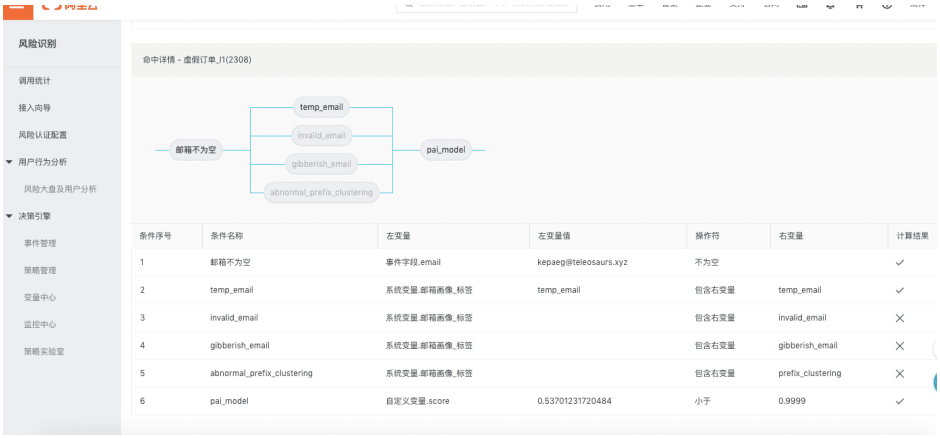
Service填写”saf_de”，代表风险识别产品

ServiceParameters可以写服务调用的时候的请求信息，本案例为：

```
{"eventCode":"de_atkpxx0397","email":"kepaeg@teleosaurs.xyz","price":"703","amount":"5","accountId":"11767964",
"currency":"CNY","f_1":"63","f_2":"100","f_3":"49"}
```



可以实时拿到请求结果，上图标明这个订单交易是有风险的。具体详细的信息，可以复制RequestID，然后进入风险识别产品的“策略实验室”中的“策略还原”，将RequestID复制进去可以看到这个订单是命中了哪种风险策略，



通过报表可以了解到，在这个订单判断中，满足了邮箱不为空，且邮箱为无效邮箱，且PAI生成的评分模型判别为高风险这3条风险判断策略。

基于外卖评论的舆情风控

业务背景

目前许多商家都有线上留言或者评论反馈平台，消费者可以在这些平台上通过留言表达自己对于消费商品的反馈。消费者的反馈包括表扬性的正向反馈，也有一些批评性质的负向反馈。商家需要掌握消费者对于产品的整体舆论取向来判断自己的产品质量是否符合消费者需求，同时了解评论内容可以方便商家分析舆论导向，指导

下一步产品研发工作。

业务痛点

目前许多酒店、餐饮、零售的留言平台每天都有大量的留言产生，传统的舆论情绪收集方案是通过人工统计的方式，但是这种方式较为低效，很难针对大规模的舆论做出精确统计。需要自动化的手段收集并判断留言平台的舆论走向。

解决方案

PAI平台提供了一套基于文本向量化以及分类的算法，可以基于历史标记的正负留言内容生成分类模型，自动对平台上的新增留言进行预测。该服务的整体框架已经基于PAI-Studio开发完成，基于真实的打标后的11987条外卖平台评论数据，实现了自动化的正反面舆论风控，准确性达到75%左右。

1.人力要求：需要具备基础的NLP及分类算法知识用于模型调试

2.开发周期：1-2天

3.数据要求：最好有超过千条的打标数据，数据越多效果越好

数据说明

序号	label	review
29	1	这次的麻辣教父一点也的不辣诶。。。不知道为啥。。。
30	1	真的是太好吃了太帅了吃的我美美的送餐也很快以后外卖就百度这家餐厅了
31	1	今天的牛肉烧烤饭，感觉牛肉有些不新鲜，送餐员的速度还是很快的。
32	1	大雾霾，外卖小哥记得带口罩哦！给爸妈带回天津尝尝稻香村的的小肚～～～
33	1	"棒棒哒棒棒哒棒棒哒,师傅辛苦"
34	1	挺好的，不错
35	1	"外卖速度快,饭菜依然好吃,点赞"
36	1	饭菜很好吃
37	1	前几天点的卤肉饭要是单卖里面的泡菜就更好了
38	0	"糟糕,继续努力吧"
39	0	这个很难吃
40	0	菜明显是剩菜，跟之前买的完全不一样
41	0	卷饼不错，但等了两个小时，什么情况
42	0	晚上七点订的外卖，九点还没送到，电话说是忘了我的订单，说好的退款一直没有退还
43	0	卷饼的量太小了，，
44	0	因为楼层很多所以让人去校门口自取，好懒……
45	0	香菇鸡肉不太好吃，果汁也太袖珍了吧…不过速度巨快

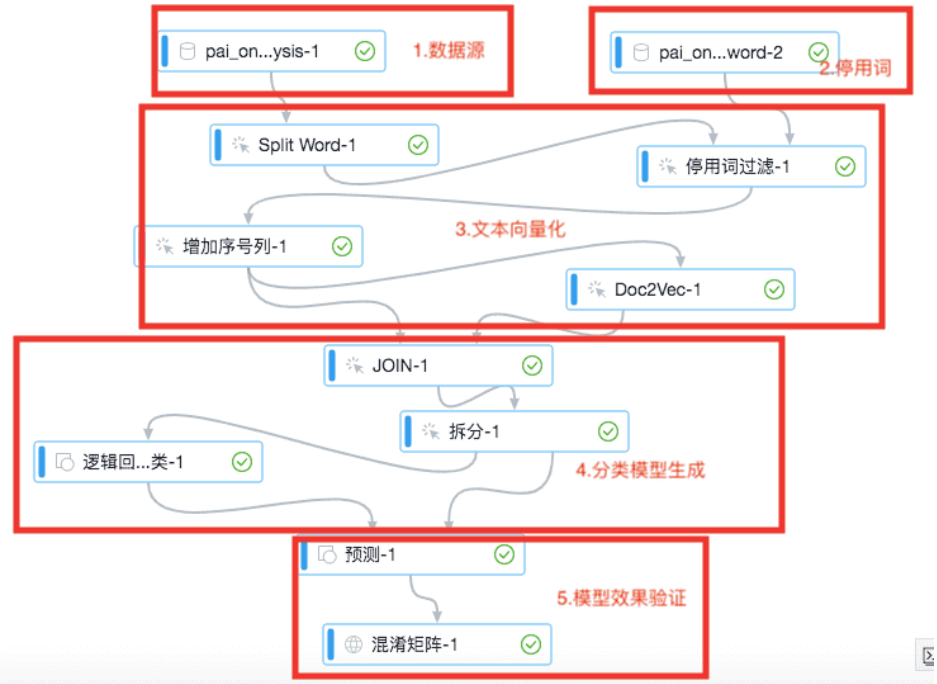
参数名称	参数描述
label	标签，1是正向评论，0为负面评论
review	实际评论数据

流程说明

进入PAI-Studio产品：https://pai.data.aliyun.com/console

该方案数据和实验环境已经内置于首页模板：

打开实验：



1.数据源

上文提到的评论数据

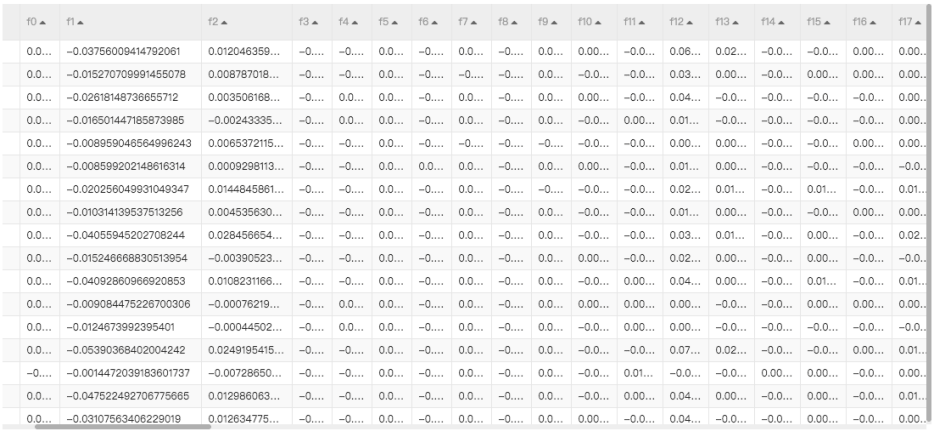
2.停用词

过滤一些助动词以及标点符号，需要自己上传停用词表

序号	stop_word
9	7
10	8
11	9
12	?
13	_
14	“
15	”
16	、
17	。
18	《
19	》

3.文本向量化

利用Doc2vector算法把每个评论变成语意向量，每行是一个向量，每个向量代表一个评论的含义



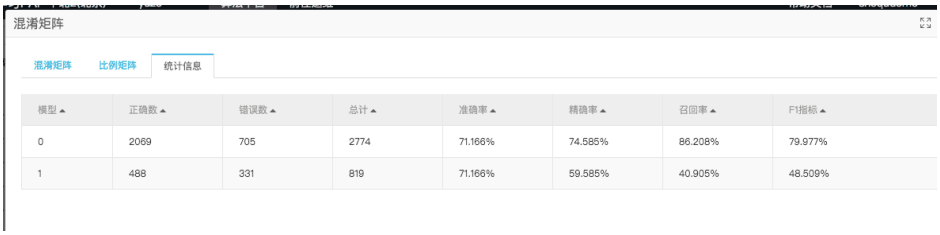
f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15	f16	f17
0.0...	−0.03756009414792061	0.012046359...	−0....	−0....	0.0...	−0....	0.0...	−0....	0.0...	0.00...	−0.0...	0.06...	0.02...	−0.0...	−0.0...	0.00...	0.00...
0.0...	−0.015270709991455078	0.008787018...	−0....	−0....	0.0...	−0....	−0....	−0....	0.0...	−0.0...	−0.0...	0.03...	0.00...	−0.0...	0.00...	0.00...	0.00...
0.0...	−0.026181487366557 12	0.003506168...	−0....	0.0...	0.0...	−0....	0.0...	−0....	0.0...	0.00...	−0.0...	0.04...	−0.0...	−0.0...	−0.0...	−0.0...	0.00...
0.0...	−0.016501447185873985	−0.00243335...	−0....	0.0...	0.0...	−0....	0.0...	−0....	0.0...	−0.0...	0.00...	0.01...	−0.0...	−0.0...	−0.0...	−0.0...	0.00...
0.0...	−0.008959046564996243	0.0065372115...	−0....	−0....	0.0...	−0....	−0....	−0....	−0....	−0.0...	−0.0...	0.00...	0.00...	−0.0...	−0.0...	0.00...	0.00...
0.0...	−0.008599202148616314	0.0009298113...	−0....	−0....	0.0...	0.0...	0.0...	−0....	0.0...	0.00...	−0.0...	0.01...	0.00...	−0.0...	−0.0...	−0.0...	−0.0...
0.0...	−0.020256049931049347	0.0144845861...	−0....	−0....	0.0...	−0....	0.0...	−0....	−0....	−0.0...	−0.0...	0.02...	0.01...	−0.0...	0.01...	−0.0...	0.01...
0.0...	−0.010314139537513256	0.004535630...	−0....	−0....	0.0...	−0....	0.0...	−0....	0.0...	−0.0...	−0.0...	0.01...	0.00...	−0.0...	−0.0...	0.00...	0.00...
0.0...	−0.04055945202708244	0.028456654...	−0....	−0....	0.0...	−0....	0.0...	−0....	0.0...	−0.0...	−0.0...	0.03...	0.01...	−0.0...	0.00...	−0.0...	0.02...
0.0...	−0.015246668830513954	−0.00390523...	−0....	−0....	0.0...	−0....	0.0...	−0....	0.0...	0.00...	−0.0...	0.02...	0.00...	−0.0...	0.00...	−0.0...	−0.0...
0.0...	−0.04092860966920853	0.0108231166...	−0....	−0....	0.0...	−0....	0.0...	−0....	0.0...	−0.0...	0.00...	0.04...	0.00...	−0.0...	0.01...	−0.0...	0.01...
0.0...	−0.009084475226700306	−0.00076219...	−0....	0.0...	0.0...	−0....	0.0...	−0....	0.0...	0.00...	0.00...	0.00...	−0.0...	−0.0...	0.00...	0.00...	0.00...
0.0...	−0.0124673992395401	−0.00044502...	−0....	0.0...	0.0...	−0....	0.0...	−0....	0.0...	−0.0...	0.00...	0.00...	−0.0...	−0.0...	−0.0...	−0.0...	−0.0...
0.0...	−0.05390368402004242	0.0249195415...	−0....	−0....	0.0...	−0....	0.0...	−0....	0.0...	−0.0...	−0.0...	0.07...	0.02...	−0.0...	−0.0...	0.00...	0.01...
−0....	−0.0014472039183601737	−0.00728650...	−0....	−0....	0.0...	−0....	0.0...	−0....	0.0...	−0.0...	0.01...	−0.0...	−0.0...	0.00...	0.00...	−0.0...	0.00...
0.0...	−0.047522492706775665	0.012986063...	−0....	−0....	0.0...	−0....	0.0...	−0....	0.0...	−0.0...	0.00...	0.04...	0.00...	−0.0...	0.00...	−0.0...	0.01...
0.0...	−0.03107563406229019	0.012634775...	−0....	−0....	0.0...	−0....	0.0...	−0....	0.0...	0.00...	−0.0...	0.04...	−0.0...	−0.0...	0.00...	−0.0...	0.00...

4.生成分类模型

将向量化后的文本通过拆分算法拆分为训练集以及测试集，训练集通过逻辑回归算法训练生成二分类模型，该模型可以实现对于评论是正向评论或者负向评论的判断。

5.模型效果验证

通过混淆矩阵算法验证模型的实际效果，


混淆矩阵

混淆矩阵 比例矩阵 统计信息

模型	正确数	错误数	总计	准确率	精确率	召回率	F1指标
0	2069	705	2774	71.166%	74.585%	86.208%	79.977%
1	488	331	819	71.166%	59.585%	40.905%	48.509%

总结

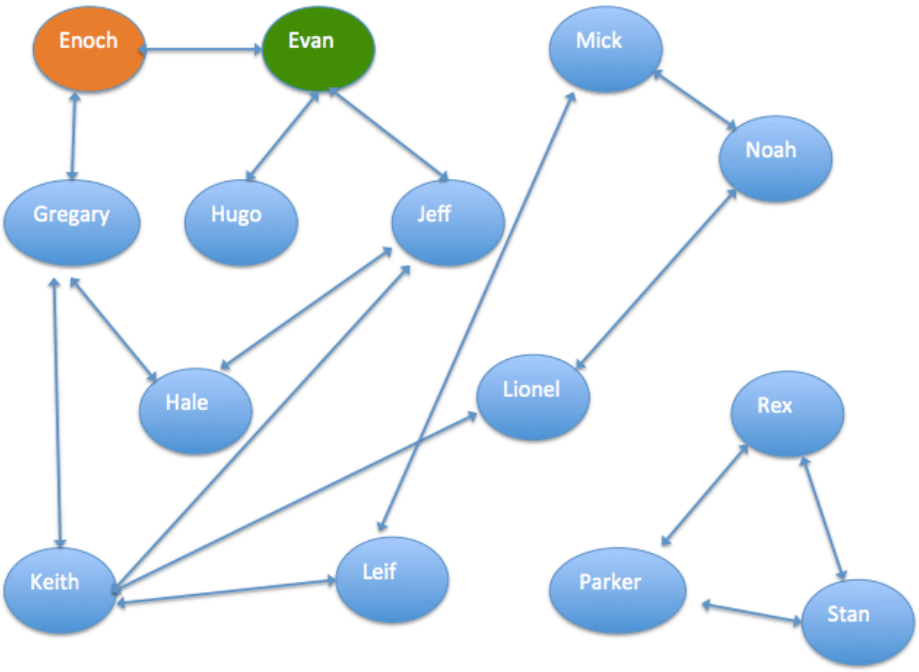
本文提到的基于留言评论的舆情风控方案可以用PAI组件在1-2天时间内非常快速的实现，实现后可以批量的对于平台上面的留言舆论进行智能化分析，并且随着数据的累计，模型的准确性会逐渐增强。该方案适用用各种基于文本场景的分析，比如垃圾邮件分类、新闻正反情绪分类等。

【图算法】金融风控实验

本文数据为虚构，仅供实验。

背景

图算法一般用来解决关系网状的业务场景。与常规的结构化数据不同，图算法需要把数据整理成首尾相连的关系图谱，更多考虑的是边和点的概念。阿里云机器学习平台上提供了丰富的图算法组件，包括K-Core、最大联通子图、标签传播聚类等。
本文档针对阿里云机器学习平台上图算法模块来进行实验，业务场景如下。
下图是已知的一份人物通联关系图，每两个人之间的连线表示两人有一定关系，可以是同事或者亲人关系等。已知“Enoch”是信用用户，“Evan”是欺诈用户。需要通过图算法，计算出其它人的信用指数，即得到图中每个人是欺诈用户的概率。这个数据可以方便相关机构做风控。



数据集介绍

具体字段如下表所示。

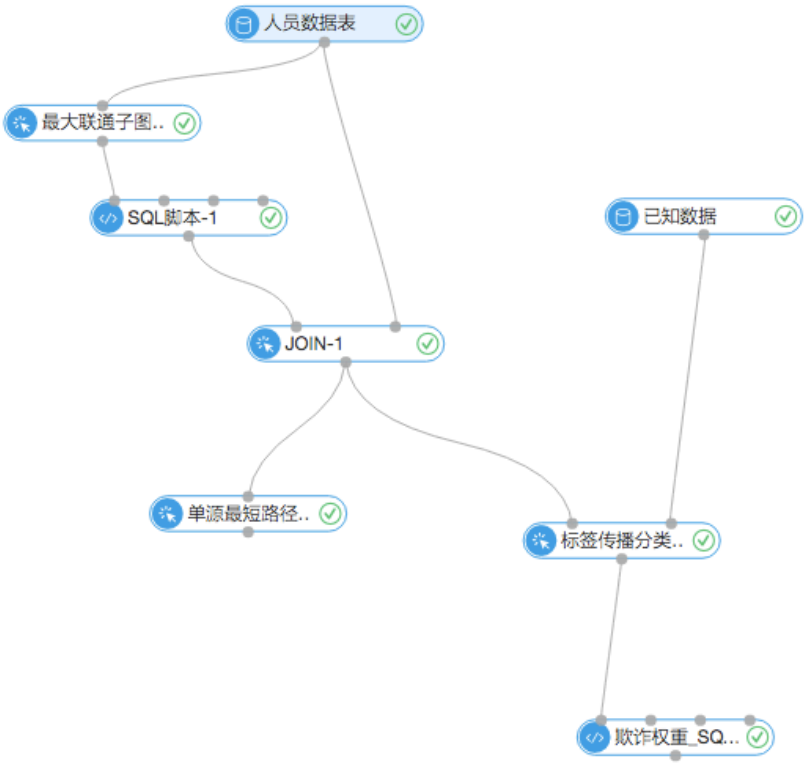
字段名	含义	类型	描述
start_point	边的起始节点	string	人
end_point	边结束节点	string	人
count	关系紧密度	double	数值越大，两人的关系越紧密

数据截图如下。

start_point ▲	end_point ▲	count ▲
Enoch	Evan	10
Enoch	Gregary	2
Gregary	Hale	6
Evan	Hugo	2
Evan	Jeff	4
Gregary	Keith	7
Jeff	Keith	5
Hale	Jeff	11
Keith	Leif	3
Keith	Lionel	1
Leif	Mick	4

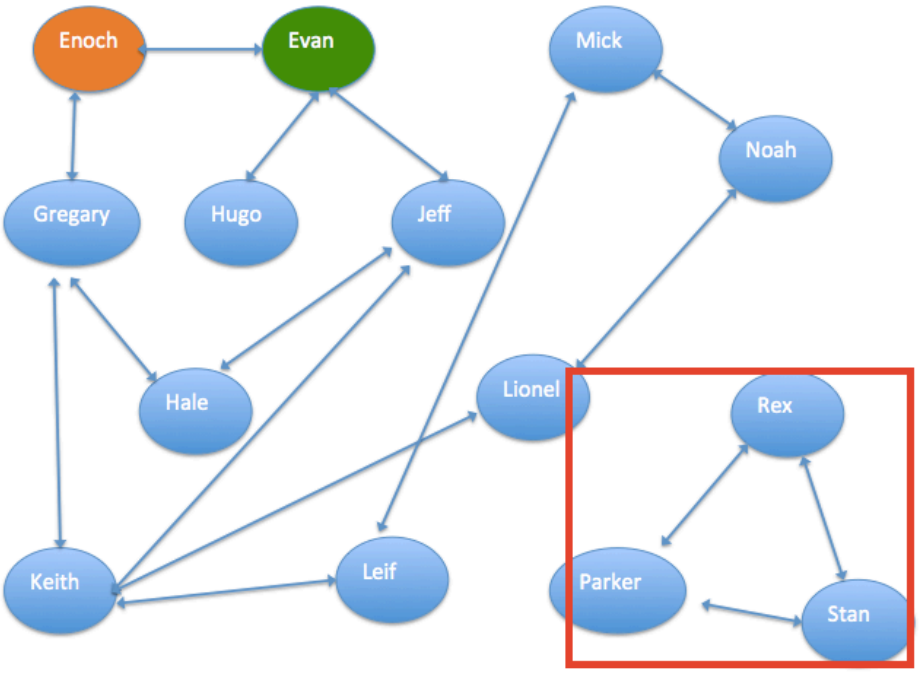
数据探索流程

实验流程图如下。



1. 最大联通子图

最大联通子图的功能：图算法的输入数据是关系图谱结构的，最大联通子图可以找到有通联关系的最大集合，在团伙发现的场景中可以排除掉一些与风控场景无关的人。
本次实验通过**最大联通子图**组件将数据中的群体分为两部分，并赋予group_id。通过**SQL脚本**组件和**JOIN**组件去除下图中的无关联人员。



2. 单源最短路径

通过**单源最短路径**组件探查出每个人的一度人脉、二度人脉等关系。“distance”表示“Enoch”通过几个人可以联络到目标人，如下图所示。

start_node ▲	dest_node ▲	distance ▲	distance_cnt ▲
Enoch	Hale	2	1
Enoch	Leif	3	1
Enoch	Hugo	2	1
Enoch	Keith	2	1
Enoch	Jeff	2	1
Enoch	Evan	1	1
Enoch	Lionel	3	1
Enoch	Mick	4	1
Enoch	Gregary	1	1
Enoch	Noah	4	1
Enoch	Enoch	0	0

3. 标签传播分类

标签传播分类算法为半监督的分类算法，原理是用已标记节点的标签信息去预测未标记节点的标签信息。在算

法执行过程中，每个节点的标签按相似度传播给相邻节点。
使用**标签传播分类**组件除了需要所有人员的通联图数据以外，还要有人员打标数据。本实验通过**已知数据（读数据表）**组件导入打标数据（ “weight”表示目标是欺诈用户的概率），如下图所示。

point	point_type	weight
Enoch	信用用户	1
Evan	欺诈用户	0.8

4. 结论

通过**SQL脚本**组件对结果进行筛选，最终展现的是每个人涉嫌欺诈的概率，数值越大表示是欺诈用户的概率越大，如下图所示。

node	tag	weight
Hugo	欺诈用户	1
Evan	欺诈用户	0.8
Noah	欺诈用户	0.42059743476528927
Jeff	欺诈用户	0.34784053907648443
Mick	欺诈用户	0.3113287445872401
Lionel	欺诈用户	0.2938277295951075
Leif	欺诈用户	0.24091136964145973
Keith	欺诈用户	0.2264783897173419

评分卡信用评分

机器学习算法基于信用卡消费记录做信用评分

背景

评分卡是**信用风险评估**和**互联网金融**领域常用的建模方法，并不简单对应于某一种机器学习算法，而是一种通用的建模框架。它将原始数据通过分箱后进行特征工程变换，继而应用于线性模型进行建模。

评分卡建模理论常被用于各种信用评估领域，比如信用卡风险评估、贷款发放等业务。另外，在其它领域评分卡常被用来作为分数评估，比如常见的客服质量打分、芝麻信用分打分等。本文档通过一个案例讲解如何通过机器学习平台的金融板块组件，搭建出一套评分卡建模方案。

在首页可以找到该模板，如果没有看到，单击右下角**加载更多**，可以直接从模板创建评分卡实验，如下图所示。该模板包含了整个实验的流程和数据。

数据集介绍

源表字段信息

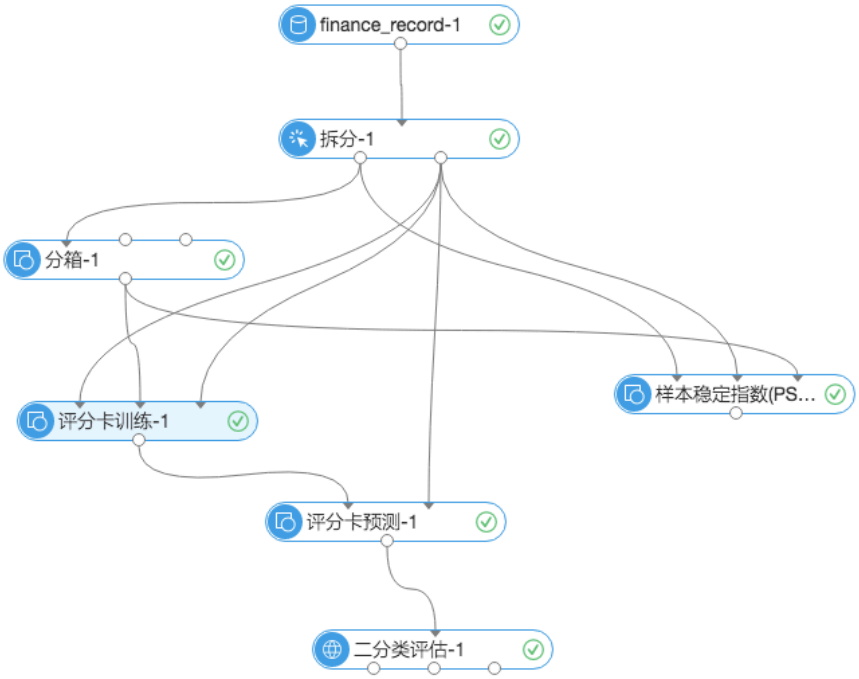
字段	类型	前 100 条记录
id	STRING	1,2,3,4,5
limit_bal	BIGINT	20000,50000,
sex	STRING	女,男
education	STRING	本科
marriage	STRING	已婚,未婚
age	BIGINT	24,26,34,37,5
pay_0	BIGINT	-1,0,2
pay_2	BIGINT	0,2
pay_3	BIGINT	-1,0
pay_4	BIGINT	-1,0
pay_5	BIGINT	-2,0
pay_6	BIGINT	-2,0,2
bill_amt1	DOUBLE	2682.0,3913.(
bill_amt2	DOUBLE	1725.0,3102.(
bill_amt3	DOUBLE	689.0,2682.0,
bill_amt4	DOUBLE	0.0,3272.0,14
bill_amt5	DOUBLE	0.0,3455.0,14
bill_amt6	DOUBLE	0.0,3261.0,15
pay_amt1	DOUBLE	0.0,1518.0,20
pay_amt2	DOUBLE	689.0,1000.0,
pay_amt3	DOUBLE	0.0,1000.0,12
pay_amt4	DOUBLE	0.0,1000.0,11
pay_amt5	DOUBLE	0.0,689.0,100

上图中是一份国外某机构开源的数据集，共30000条。包含了每个用户的性别、教育、婚姻、年龄等属性，及用户过去一段时间的信用卡消费情况和账单情况。payment_next_month是目标队列，表示用户是否偿还信用卡账单，1表示偿还，0表示没有偿还。

数据集下载地址：https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset

实验流程

实验流程图如下。



拆分
将输入数据集分为两部分，一部分用来训练模型，另一部分用来预测评估。

分箱
分箱组件类似于onehot编码，可以将数据按照分布映射成更高维度的特征。以age字段为例，分箱组件可以按照数据在不同区间的分布进行分箱操作，分箱结果如图所示。

	Index	Label	Constraint		WoE		Number			Rate		
			Operator	Value	WoE	Chart	Total	Positive	Negative	Total	Positive	Negative
	0	(-inf,25]			0.249		3082	822	2260	12.84%	15.5%	12.09%
	1	(25,27]			-0.12		2184	439	1745	9.1%	8.28%	9.33%
	2	(27,29]			-0.137		2421	480	1941	10.09%	9.05%	10.38%
	3	(29,31]			-0.196		2084	394	1690	8.68%	7.43%	9.04%
	4	(31,34]			-0.2		2791	526	2265	11.63%	9.92%	12.11%
	5	(34,37]			-0.016		2622	572	2050	10.93%	10.79%	10.96%
	6	(37,40]			-0.025		2224	482	1742	9.27%	9.09%	9.32%
	7	(40,43]			0.026		1823	411	1412	7.6%	7.75%	7.55%
	8	(43,49]			0.083		2628	619	2009	10.95%	11.67%	10.74%
	9	(49,+inf)			0.215		2141	557	1584	8.92%	10.51%	8.47%
	-2	ELSE					-	-	-	-	-	-

最终分箱组件的输出如下图所示，每个字段都被分箱到多个区间上。

序号	feature	json
1	limit_bal	{ "bin": {"norm": [{ "iv": 0.076602, "n": 2104, "p": 1187, "prate": 0.360681, "total": 3291, "value": "(-inf,30000]", "woe": 0.687921}, { "iv": 0.009549999999999999, "n": 2095...
2	age	{ "bin": {"norm": [{ "iv": 0.008506, "n": 2260, "p": 822, "prate": 0.26671, "total": 3082, "value": "(-inf,25]", "woe": 0.248953}, { "iv": 0.00126, "n": 1745, "p": 439, "prate": 0.2...
3	pay_0	{ "bin": {"norm": [{ "iv": 0.047172, "n": 5735, "p": 1052, "prate": 0.155002, "total": 6787, "value": "(-inf,-1]", "woe": -0.435562}, { "iv": 0.170225, "n": 10252, "p": 1518, "prat...
4	pay_2	{ "bin": {"norm": [{ "iv": 0.007479, "n": 2483, "p": 547, "prate": 0.180526, "total": 3030, "value": "(-inf,-2]", "woe": -0.252442}, { "iv": 0.028735, "n": 4094, "p": 779, "prate": ...
5	pay_3	{ "bin": {"norm": [{ "iv": 0.006939, "n": 2676, "p": 601, "prate": 0.183399, "total": 3277, "value": "(-inf,-2]", "woe": -0.233151}, { "iv": 0.032692, "n": 4040, "p": 744, "prate": ...
6	pay_4	{ "bin": {"norm": [{ "iv": 0.004796, "n": 2826, "p": 665, "prate": 0.19049, "total": 3491, "value": "(-inf,-2]", "woe": -0.186498}, { "iv": 0.02676, "n": 3858, "p": 736, "prate": 0.1...
7	pay_5	{ "bin": {"norm": [{ "iv": 0.003088, "n": 2925, "p": 717, "prate": 0.19687, "total": 3642, "value": "(-inf,-2]", "woe": -0.145641}, { "iv": 0.023437, "n": 3740, "p": 729, "prate": 0....
8	pay_6	{ "bin": {"norm": [{ "iv": 0.002296, "n": 3135, "p": 788, "prate": 0.200867, "total": 3923, "value": "(-inf,-2]", "woe": -0.120554}, { "iv": 0.019253, "n": 3847, "p": 783, "prate": ...
9	bill_amt1	{ "bin": {"norm": [{ "iv": 0.001611, "n": 1818, "p": 584, "prate": 0.243131, "total": 2402, "value": "(-inf,282]", "woe": 0.124741}, { "iv": 3e-06, "n": 1866, "p": 532, "prate": 0.2...
10	bill_amt2	{ "bin": {"norm": [{ "iv": 0.000701, "n": 1929, "p": 593, "prate": 0.235131, "total": 2522, "value": "(-inf,0]", "woe": 0.08076999999999999}, { "iv": 0, "n": 1789, "p": 508, "prat...
11	bill_amt3	{ "bin": {"norm": [{ "iv": 0.000503, "n": 2158, "p": 653, "prate": 0.232302, "total": 2811, "value": "(-inf,0]", "woe": 0.064972}, { "iv": 5.2e-05, "n": 1541, "p": 448, "prate": 0.2...
12	bill_amt4	{ "bin": {"norm": [{ "iv": 0.000712, "n": 2362, "p": 721, "prate": 0.233863, "total": 3083, "value": "(-inf,0]", "woe": 0.073708}, { "iv": 0.000344, "n": 1317, "p": 400, "prate": 0....
13	bill_amt5	{ "bin": {"norm": [{ "iv": 0.001599, "n": 2535, "p": 799, "prate": 0.239652, "total": 3334, "value": "(-inf,0]", "woe": 0.105744}, { "iv": 2.4e-05, "n": 1141, "p": 330, "prate": 0.2...
14	bill_amt6	{ "bin": {"norm": [{ "iv": 0.0002, "n": 2917, "p": 857, "prate": 0.22708, "total": 3774, "value": "(-inf,0]", "woe": 0.035459}, { "iv": 0.000112, "n": 791, "p": 236, "prate": 0.2297...
15	pay_amt1	{ "bin": {"norm": [{ "iv": 0.098387, "n": 2681, "p": 1516, "prate": 0.36121, "total": 4197, "value": "(-inf,0]", "woe": 0.690218}, { "iv": 0.000189, "n": 463, "p": 143, "prate": 0.2...
16	pay_amt2	{ "bin": {"norm": [{ "iv": 0.068019, "n": 2864, "p": 1441, "prate": 0.334727, "total": 4305, "value": "(-inf,0]", "woe": 0.573451}, { "iv": 0.002296, "n": 356, "p": 139, "prate": 0....
17	pay_amt3	{ "bin": {"norm": [{ "iv": 0.061212, "n": 3232, "p": 1541, "prate": 0.322858, "total": 4773, "value": "(-inf,0]", "woe": 0.519663}, { "iv": 7.7e-05, "n": 31, "p": 7, "prate": 0.1842...

样本稳定指数PSI

样本稳定指数是衡量样本变化所产生的偏移量的一种重要指标，通常用来衡量样本的稳定程度。比如样本在两个月份之间的变化是否稳定。通常变量的PSI值在0.1以下表示变化不太显著，在0.1到0.25之间表示变化比较显著，大于0.25表示变量变化比较剧烈，需要特殊关注。

本案例中，综合比较拆分前后以及分箱结果的样本稳定程度，返回每个特征的PSI数值，如下图所示

收起 展开

Feature	Bin	Test %	Base %	Test - Base	ln(Test/Base)	PSI
limit_bal	-	-	-	-	-	0.0019
age	-	-	-	-	-	0.0005
pay_0	-	-	-	-	-	0.0002
pay_2	-	-	-	-	-	0.0006
pay_3	-	-	-	-	-	0.0005
pay_4	-	-	-	-	-	0.0016
pay_5	-	-	-	-	-	0.0015
pay_6	-	-	-	-	-	0.0019
bill_amt1	-	-	-	-	-	0.001
bill_amt2	-	-	-	-	-	0.0025
bill_amt3	-	-	-	-	-	0.0022
bill_amt4	-	-	-	-	-	0.0014
bill_amt5	-	-	-	-	-	0.0011
bill_amt6	-	-	-	-	-	0.0009
pay_amt1	-	-	-	-	-	0.0032
pay_amt2	-	-	-	-	-	0.0009

评分卡训练

评分卡训练的结果图如下所示。

	Variable	Selected	Bin Id	Variable/Bin	Const.	Weight		WOE	Importance	Train				
						Unscaled	Scaled			Total	Positive	Negative	% Pos	% Neg
	intercept	-	-	-	-	-1.254	531	-	-	-	-	-	-	-
⊟	pay_0	✓	-	-	-	0.789	-	-	4.445e-2	-	-	-	-	-
	---------	-	0	(-inf,-1]	-	-0.34	-20	-0.415	-	1648	266	1382	19.65	28.75
	---------	-	1	(-1,0]	-	-0.51	-29	-0.706	-	2943	370	2573	27.33	55.38
	---------	-	2	(0,1]	-	0.474	27	0.562	-	757	256	501	18.91	10.78
	---------	-	3	(1,2]	-	1.618	93	2.12	-	562	398	164	29.39	3.53
	---------	-	4	(2,+inf)	-	1.747	101	2.134	-	90	64	26	4.73	0.56
	---------	-	-2	ELSE	-	0	0	-	-	0	0	0	0	0
	---------	-	-1	NULL	-	0	0	-	-	0	0	0	0	0
⊟	limit_bal	✓	-	-	-	0.453	-	-	2.414e-3	-	-	-	-	-
	---------	-	0	(-inf,30000]	-	0.299	17	0.743	-	803	305	498	22.53	10.72
	---------	-	1	(30000,50000]	-	0.124	7	0.269	-	710	196	514	14.48	11.06
	---------	-	2	(50000,70000]	-	0.168	10	0.208	-	337	89	248	6.57	5.34
	---------	-	3	(70000,100000]	-	0.058	3	0.161	-	639	163	476	12.04	10.25
	---------	-	4	(100000,140...	-	0.02	1	0.033	-	579	134	445	9.9	9.58
	---------	-	5	(140000,180...	-	-0.126	-7	-0.398	-	684	112	572	8.27	12.31
	---------	-	6	(180000,210...	-	-0.139	-8	-0.222	-	486	92	394	6.79	8.48

评分卡的精髓是将复杂的模型权重用符合业务标准的分数表示。

- intercepy：截距。
- Unscaled：原始的权重值。
- Scaled：分数更改指标，比如对于pay_0这个特征，如果特征落在(-1,0]之间分数就减29，如果特征落在（0，1]之间分数就加上27。
- importance：每个特征对于结果的影响大小，数值越大表示影响越大。

评分卡预测

每个预测结果的最终评分，本案例中表示的是每个用户的信用评分。

序号	payment_next_month	prediction_score	prediction_prob	prediction_detail
1	0	499	0.14314626458020613	{"0":0.8568537354,"1":0.1431462646}
2	0	564	0.3367775480162267	{"0":0.6632224520,"1":0.3367775480}
3	0	555	0.3035873747480541	{"0":0.6964126253,"1":0.3035873747}
4	1	519	0.18818103244164777	{"0":0.8118189676,"1":0.1881810324}
5	1	651	0.7013570482913543	{"0":0.2986429517,"1":0.7013570483}
6	0	502	0.1474992646536902	{"0":0.8525007353,"1":0.1474992647}
7	1	560	0.3199046397072833	{"0":0.6800953603,"1":0.3199046397}
8	0	435	0.05207880036730361	{"0":0.9479211996,"1":0.0520788004}
9	0	491	0.12535852489673346	{"0":0.8746414751,"1":0.1253585249}

结论

基于用户的信用卡消费记录，通过评分卡模型训练及评分卡预测得到了每个用户的最终信用评分，这个评分可以应用到各种贷款或者金融相关的征信领域中。

异常指标监控

业务背景

用户系统中如果出现任何的异常数据，比如一个运维系统的CPU消耗突然增高，比如平台突然有大量不良信息

产生，比如有用户大量薅羊毛，这些行为都是平台的异常指标。如果能通过机器学习的方式帮助用户针对各种异常指标做预防和实时预警，将大大建设平台方的风险。

业务痛点

缺乏一种实时高效的方式监控平台指标，增强平台的智能化安全防卫能力。

解决方案

PAI平台提供了一套基于指标监控的分类算法，可以把异常指标监控抽象为一个二分类场景，并且把监控模型部署到在线系统实时调用，实现近线风控。

1.人力要求：需要懂机器学习经典算法特别是特征工程以及二分类算法的同学

2.开发周期：1-2天

3.数据要求：已经达标过的数据上千条，标记出哪些数据是异常数据，哪些是非异常数据

数据说明

文案例使用的数据是一份系统级别监控日志数据，一共22544条数据，其中异常数据9711条。

service	flage	a2	a3	a4	a5	a6	a7	a8	a9	a10	a11	a12	a13	a14	a15	a16	a17	a18	a19	a2
private	REJ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	22
private	REJ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13
ftp_data	SF	12...	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eco_i	SF	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
telnet	RSTO	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
http	SF	267	14...	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	4
smtp	SF	1022	387	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
telnet	SF	129	174	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
http	SF	327	467	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	33
ftp	SF	26	157	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1
telnet	SF	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
smtp	SF	616	330	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
private	REJ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	111
telnet	S0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12
telnet	SF	773	36...	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
http	SF	350	3610	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	8
http	SF	213	659	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	24

数据说明：

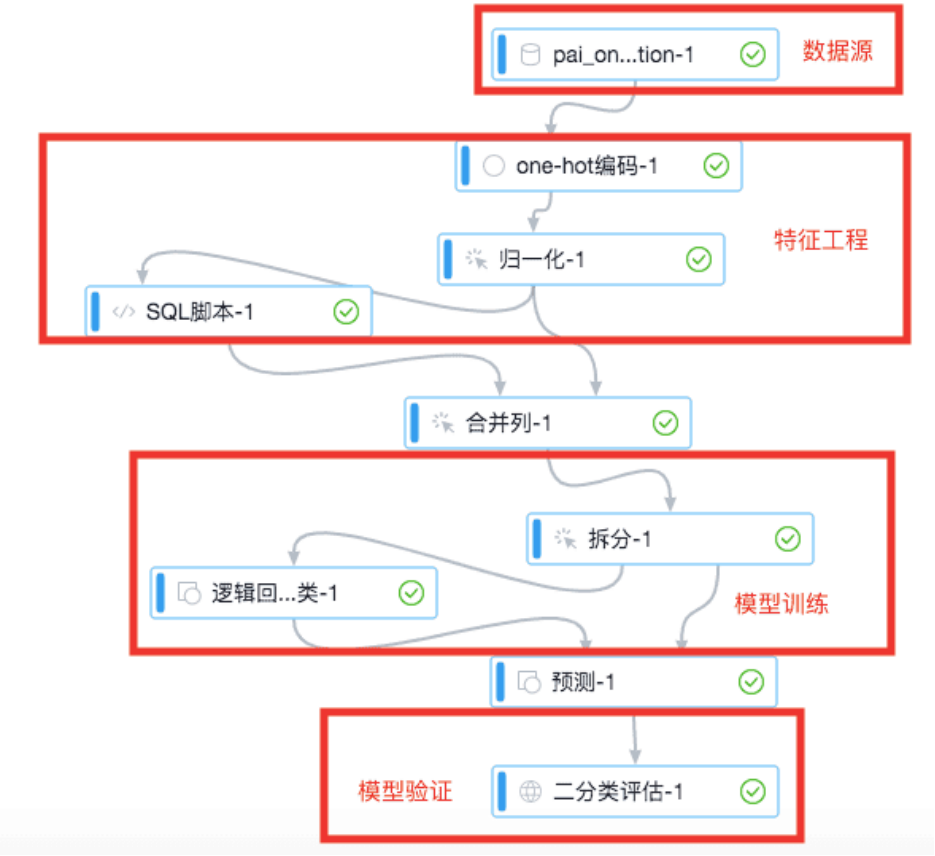
参数名称	参数描述
protocol_type	网络连接协议，有tcp、icmp、udp等
service	服务协议，有http、finger、pop、private、smtp等
flage	SF、RSTO、REJ
a2~a38	不同的一些系统指标
class	标签字段，其中normal为正常样本，anomaly为异常样本

流程说明

进入PAI-Studio产品：https://pai.data.aliyun.com/console

该方案数据和实验环境已经内置于首页模板：

打开实验：



1. 数据源

数据说明中提到的数据。

2. 特征工程

one-hot特征编码组件可以自动将特征由字符型向数值型转变，是机器学习领域最常见的数据编码方式。

归一化组件可以将所有数据的范围都限定到0~1之间，去除量纲的影响。归一化后数据如下图：

a1	a2	a3	a4	a5	a6	a7	a8	a9	a10	a11
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0.00003...	0.00020...	0	0	0	0	0	0	0	0	0
0	3.183413...	0	0	0	0	0	0	0	0	0
0.00001...	0	0.0...	0	0	0	0	0	0	0	0
0	0.00000...	0.0...	0	0	0	0	0	1	0	0
0	0.00001...	0.0...	0	0	0	0	0	1	0	0
0	0.00000...	0.0...	0	0	0	0	0.25	0	0	0
0	0.00000...	0.0...	0	0	0	0	0	1	0	0
0	4.138437...	0.0...	0	0	0	0	0.25	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0.00000...	0.0...	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0.00064...	0.00001...	0.2...	0	0	0	0	0	1	0	0
0	0.00000...	0.0...	0	0	0	0	0	1	0	0
0	0.00000...	0.0...	0	0	0	0	0	1	0	0

利用SQL组件把目标列是anomaly的标记为1，正常指标标记为0。

```
select (case class  when 'anomaly' then 1 else 0 end) as class from  ${t1};
```

3. 模型训练

根据正常和非正常样本训练监控模型是一个典型的二分类问题，使用机器学习领域中的逻辑回归二分类算法就能达到比较好的效果。

4. 模型评估

PAI平台提供二分类模型的评估组件，可以通过AUC、KS、F1Score等指标评估模型的好坏，本实验的模型预测准确率超过了90%。

总结

PAI平台提供了特征编码、模型训练、模型评估全方位的功能，只要能把平台产生的异常行为的特征抽取出来并标记，就可以基于PAI快速构建异常指标监控模型。

用户流失预警风控

业务背景

在业务发展过程中有两个重要的环节，一个是拉新，另一个是留存。如何做到用户的留存需要很多技术手段保证，一个比较重要的方式是建立用户流失模型，通过学习历史上流失用户的特点，通过机器学习的手段训练处风控模型，队可能会流式的用户进行预测，然后可以提前通过运营手段做一些用户流失的防范。

业务痛点

目前用户流失预警监控是业内主流的需求之一，但是缺少智能化的预测手段和机制。目前主流的一些预警方案都是基于一些规则的方案，对于一些潜在可能流失的用户没有很准确的发掘手段。

解决方案

PAI平台提供了一套基于打标数据的特征编码、分类模型训练、模型评估的方案。

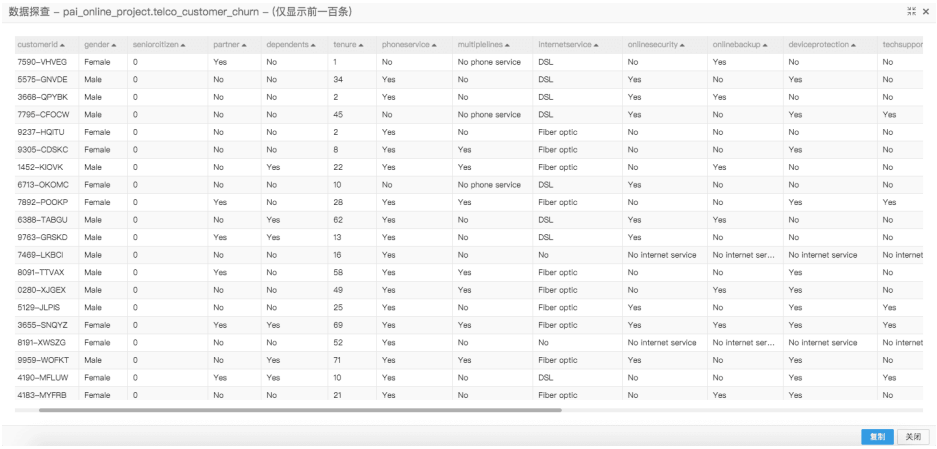
1.人力要求：需要具备基础的建模背景知识

2.开发周期：1-2天

3.数据要求：最好有超过千条的打标数据，打标哪些客户在哪种特征情况下流失过，数据越多效果越好

数据说明

数据来自真实的电信领域客户行为数据，包含用户的基本属性以及用户是否会流失，数据一共7043个用户样本。



特征数据：

参数名称	参数描述
customerid	用户ID
gender	性别

SeniorCitizen	是否是个市民，1是，0不是
Partner	是否有Partner
Dependents	是否有从属关系
tenure	客户在这个公司使用的时长
PhoneService	是否有手机服务
MultipleLine	是否有多条线路
InternetService	互联网服务商DSL、Fiber optic、No
OnlineSecurity	是否有互联网在线安全问题
OnlineBackup	是否有线上支持
DeviceProtection	是否有服务保护
TechSupport	是否申请过技术支持
StreamingTV	是否有流TV
StreamingMovies	是否有流电影
Contract	合同时限，Month-to-month、Two year
PaperlessBilling	是否有电子账单
PaymentMethod	付款方式
MonthlyCharges	月消费
TotalCharges	总消费

目标数据：

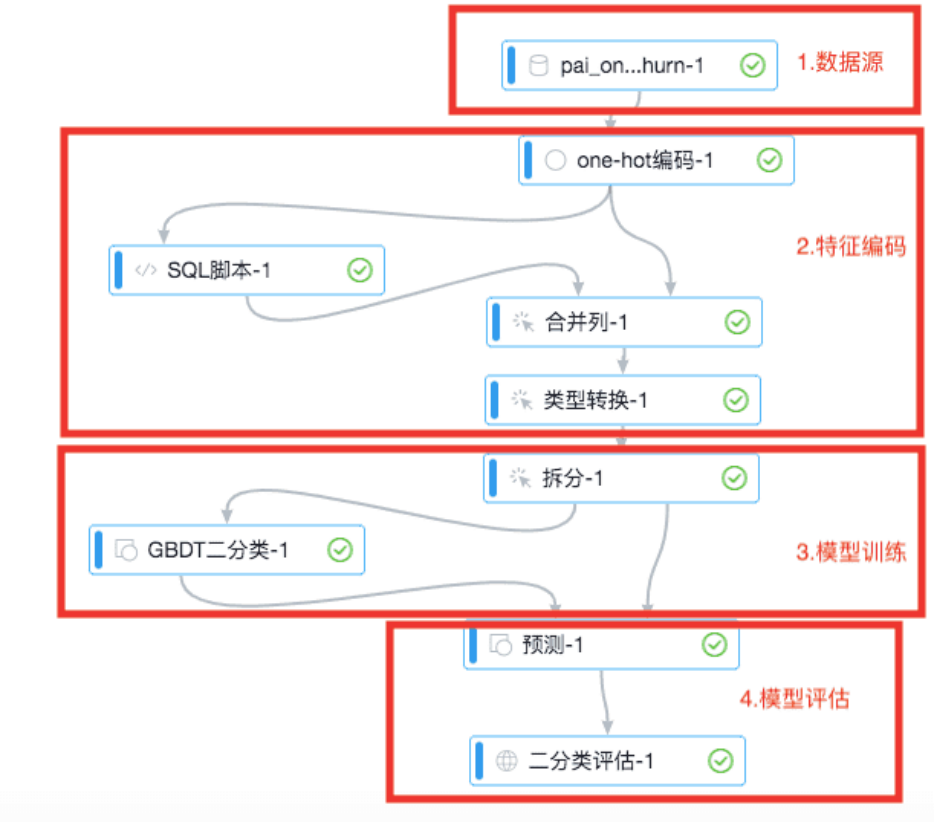
参数名称	**参数描述**
churn	用户是否流式

流程说明

进入PAI-Studio产品：https://pai.data.aliyun.com/console

该方案数据和实验环境已经内置于首页模板：

打开实验：



1.数据源

上文提到的用户流式用户的数据

2.特征编码

通过One-hot以及SQL组件实现特征工程建模，将原始的字符型特征转为数值型特征。

churn	contract_month_to_0	contract_one_year_1	contract_two_year_2	dependents_no_3	dependents_yes_4	deviceprotection_no_5	deviceprotection_no_Inter_6	deviceprotection_yes_7	gender_female_8
0	1	0	0	1	0	1	0	0	1
0	0	1	0	1	0	0	0	1	0
1	1	0	0	1	0	1	0	0	0
0	0	1	0	1	0	0	0	1	0
1	1	0	0	1	0	1	0	0	1
1	1	0	0	1	0	0	0	1	1
0	1	0	0	0	1	1	0	0	0
0	1	0	0	1	0	1	0	0	1
1	1	0	0	1	0	0	0	1	1
0	0	1	0	0	1	1	0	0	0
0	1	0	0	0	1	1	0	0	0
0	0	0	1	1	0	0	1	0	0
0	0	1	0	1	0	0	0	1	0
1	1	0	0	1	0	0	0	1	0
0	1	0	0	1	0	0	0	1	0
0	0	0	1	0	1	0	0	1	1
0	0	1	0	1	0	0	1	0	1
0	0	0	1	0	1	0	0	1	0
1	1	0	0	0	1	0	0	1	1
0	1	0	0	1	0	0	0	1	1

以目标字段churn为例，原始数据是“Yes”和“No”，可以通过SQL语句把”Yes”变为1，“No”变为0：

```
select (case churn  when 'Yes' then 1 else 0 end) as churn from  ${t1};
```

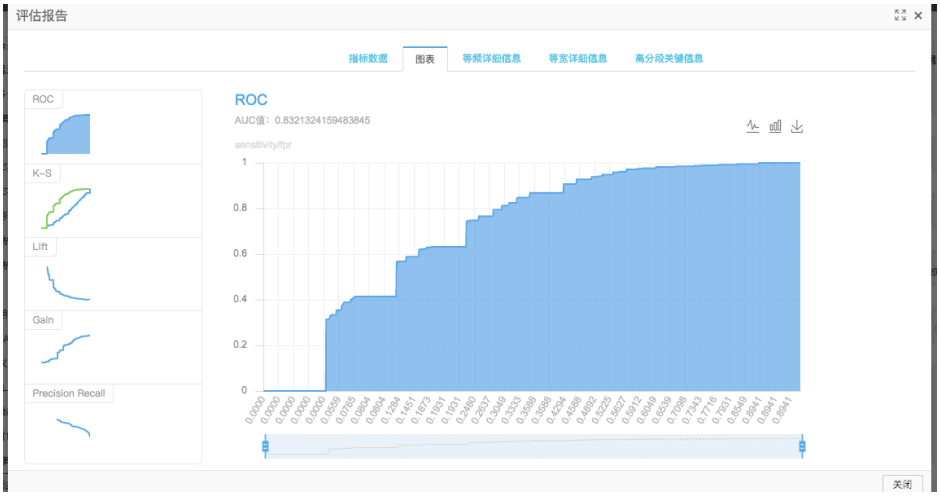
3.模型训练

将数据分成两部分，一部分作为训练集训练模型，另一部分做预测集验证模型效果。用户流失预警是个二分类

问题，一个用户只有流失和不流失两种可能性。所以选用二分类算法来处理，生成的分类模型可以一键部署为RestfulAPI服务供业务方调用。

4.模型效果验证

通过二分类评估组件验证模型准确性，准确性描述指标AUC可以达到0.83，也就是说预测的准确性在80%左右。



总结

用户流失预警是所有B端客户都可能应用到的场景，PAI提供了一套完整的基于用户特征的算法，可以帮助客户在1-2天快速实现用户流失模型的训练，大大提速了整个实验搭建的周期。>

回归算法做农业贷款发放预测

本文数据为虚构，仅供实验。

背景

农业贷款发放问题是一个典型的数据挖掘问题。贷款发放人通过往年的数据，包括贷款人的年收入、种植的作物种类、历史借贷信息等特征来构建经验模型，通过这个模型来预测受贷人的还款能力。
本文档根据真实的农业贷款业务场景，利用线性回归算法解决贷款发放业务。线性回归是利用数理统计中的回归分析方法，来确定两种或两种以上变量间相互依赖的定量关系的一种统计分析方法，运用十分广泛。本文通过农业贷款的历史发放情况，预测是否给预测集的用户发放他们所需金额的贷款。

数据集介绍

具体字段如下表所示。

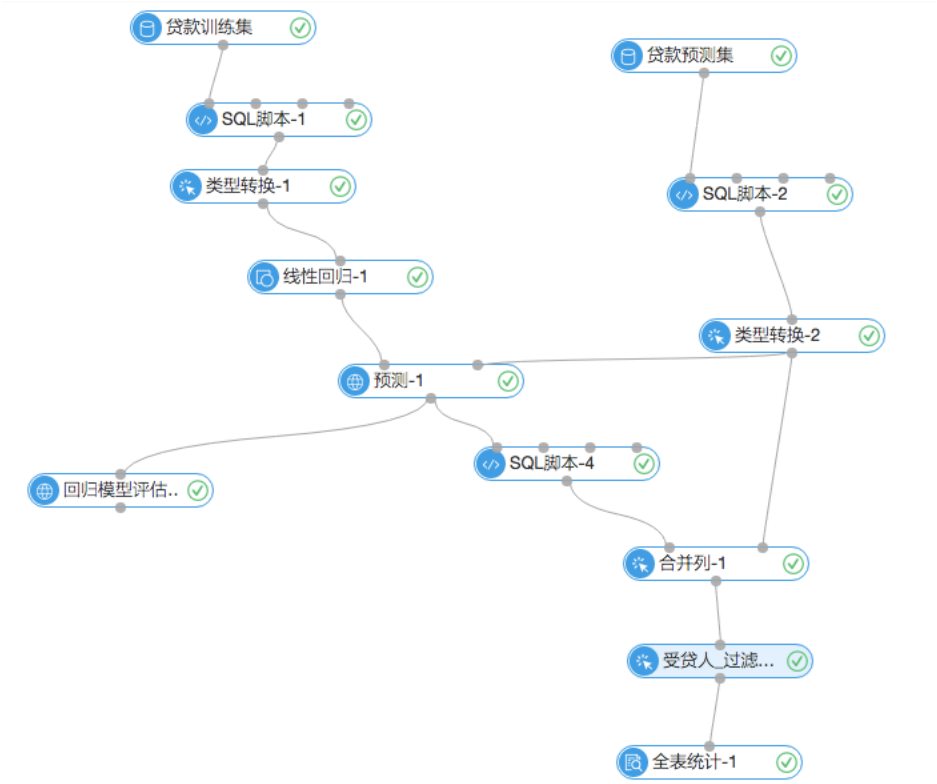
字段名	含义	类型	描述
id	数据唯一标识符	string	人
name	用户名	string	人
region	用户所属地区	string	从北到南排列
farmsize	拥有土地大小	double	土地面积
rainfall	降雨量	double	降雨量
landquality	土地质量	double	土地质量数值越大越好
farmincome	收入	double	年收入
maincrop	种植作物	string	种植作物的种类
claimtype	贷款类型	string	两种
claimvalue	贷款金额	double	贷款金额

数据截图如下。

id	name	region	farmsize	rainfall	landquality	farmincome	maincrop	claimtype	claimvalue
"id...	"name...	"midland...	1480	30	8	330729	"wheat"	"decommiss...	74703.1
"id...	"name...	"north"	1780	42	9	734118	"maize"	"arable_dev"	245354
"id...	"name...	"midland...	500	69	7	231965	"rapeseed"	"decommiss...	84213
"id...	"name...	"southw...	1860	103	3	625251	"potatoes"	"decommiss...	281082
"id...	"name...	"north"	1700	46	8	621148	"wheat"	"decommiss...	122006
"id...	"name...	"southea...	1580	42	7	445785	"maize"	"arable_dev"	122135
"id...	"name...	"southea...	1820	29	6	211605	"maize"	"arable_dev"	68969.2
"id...	"name...	"southea...	1640	108	7	1167040	"maize"	"arable_dev"	485011
"id...	"name...	"southw...	1600	101	5	756755	"wheat"	"decommiss...	160904
"id...	"name...	"southea...	600	80	6	267928	"wheat"	"arable_dev"	90350.6

数据探索流程

实验流程图如下。



1. 数据源准备

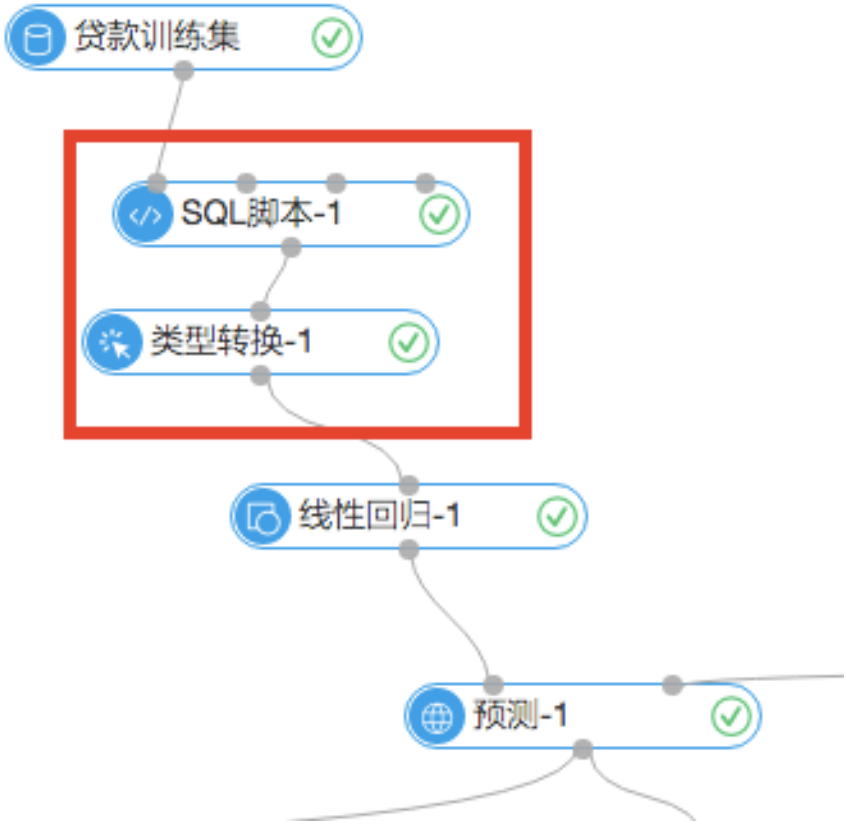
输入数据分为两部分：

- 贷款训练集：共二百余条历史贷款数据，用来训练回归模型。包括“farmsize”、“rainfall”等特征，“claimvalue”是贷款收回的金额。
- 贷款预测集：共七十一人，是今年申请贷款者，“claimvalue”是农民申请的贷款金额。

通过已有的二百余条历史数据，预测给七十一人中的哪些申请人发放贷款。

2. 数据预处理

根据含义将字符串类型的数据映射成数字。例如“region”字段，将其中的north、middle、south按照从北到南的顺序分别映射为0、1、2，再通过类型转换组件将字段转换成double类型，如下图所示。完成后即可进行模型训练。



3. 模型训练及预测

使用线性回归组件对历史数据进行训练并生成回归模型，在预测组件中利用回归模型对于预测集数据进行了预测。通过合并列组件将用户ID、预测值、申请的贷款值合并，结果如下图所示。
预测值表示的是用户的还贷能力（预期可以归还的金额）。

claimvalue	prediction_score	id
172753	164424.3413395547	1
93415.4	146370.52166158534	2
46800.2	41879.999271195346	3
131728	192648.19077439874	4
89040.8	76369.8134277192	5
135493	103695.67105783387	6
88906.8	136845.30246967232	7
147159	144156.81362150217	8
277397	466728.8170899566	9
67547.3	131340.40980772747	10
345394	402192.7992950041	11

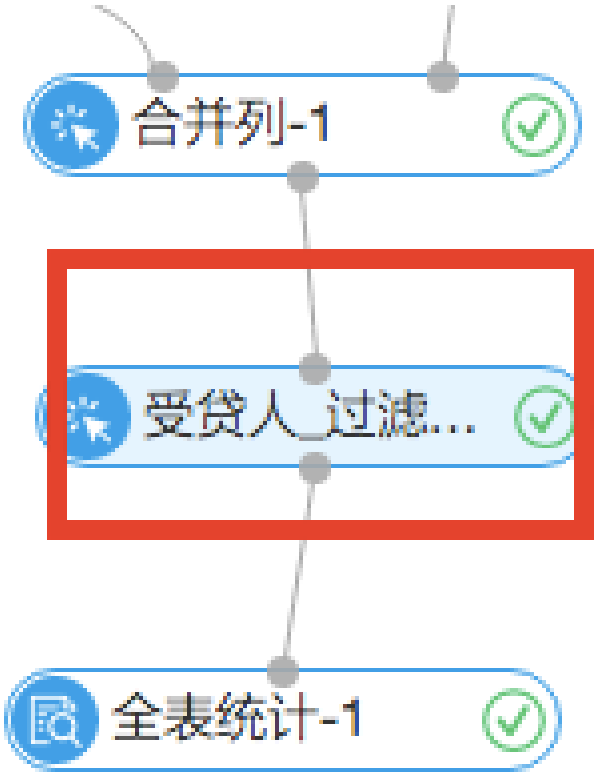
4. 回归模型评估

通过回归模型评估组件对模型进行评估，评估结果如下图所示。

字段名称	描述
SST	总平方和
SSE	误差平方和
SSR	回归平方和
R2	判定系数
R	多重相关系数
MSE	均方误差
RMSE	均方根误差
MAE	平均绝对误差
MAD	平均误差
MAPE	平均绝对百分误差
count	行数
yMean	原始因变量的均值
predictionMean	预测结果的均值

5. 贷款发放

通过过滤与映射组件筛选出可以获得贷款的人。实验的原理是针对每个客户，如果贷款人被预测得到的还款能力大于他申请贷款的金额，就给他发放贷款。



其它

请进入阿里云数加机器学习平台体验阿里云机器学习产品，并通过云栖社区公众号参与讨论。

心脏病预测案例

背景

心脏病是人类健康的头号杀手。全世界1/3的人口死亡是心脏病引起的。而我国，每年有几十万人死于心脏病。如果可以通过提取人体相关的体测指标，通过数据挖掘方式来分析不同特征对于心脏病的影响，将对预防心脏病起到至关重要的作用。本文提供真实的数据，并通过阿里云机器学习平台搭建心脏病预测案例。

数据集介绍

数据源为UCI开源数据集heart_disease。包含了303条美国某区域的心脏病检查患者的体测数据。具体字段如下表。

字段名	含义	类型	描述
age	年龄	string	对象的年龄，数字表示
sex	性别	string	对象的性别，female和male
cp	胸部疼痛类型	string	痛感由重到无typical、atypical、non-anginal、asymptomatic
trestbps	血压	string	血压数值
chol	胆固醇	string	胆固醇数值
fbs	空腹血糖	string	血糖含量大于120mg/dl为true，否则为false
restecg	心电图结果	string	是否有T波，由轻到重为norm、hyp
thalach	最大心跳数	string	最大心跳数
exang	运动时是否心绞痛	string	是否有心绞痛，true为是，false为否
oldpeak	运动相对于休息的ST depression	string	st段压数值

slop	心电图ST segment的倾斜度	string	ST segment的slope，程度分为down、flat、up
ca	透视检查看到的血管数	string	透视检查看到的血管数
thal	缺陷种类	string	并发种类，由轻到重norm、fix、rev
status	是否患病	string	是否患病，buff是健康、sick是患病

数据探索流程

数据挖掘流程如下：

整体实验流程：

1. 数据预处理

数据预处理也叫作数据清洗，主要在数据进入算法流程前对数据进行去噪、缺失值填充、类型变换等操作。本次实验的输入数据包括14个特征列和1个目标列。需要解决的问题是根据用户的体检指标预测是否会患有心脏病，每个样本只有患病或不患病两种情况，是分类问题。

本次分类实验选用的是线性模型逻辑回归，要求输入的特征都是double类型的数据，如下图所示。

数据探查 - heart_disease_prediction - (仅显示前一百条)

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slop	ca	thal	status	style
63.0	male	ang...	145.0	233.0	true	hyp	150.0	fal	2.3	down	0.0	fix	buff	H
67.0	male	asy...	160.0	286.0	fal	hyp	108.0	true	1.5	flat	3.0	norm	sick	S2
67.0	male	asy...	120.0	229.0	fal	hyp	129.0	true	2.6	flat	2.0	rev	sick	S1
37.0	male	not...	130.0	250.0	fal	norm	187.0	fal	3.5	down	0.0	norm	buff	H
41.0	fem	abn...	130.0	204.0	fal	hyp	172.0	fal	1.4	up	0.0	norm	buff	H
56.0	male	abn...	120.0	236.0	fal	norm	178.0	fal	0.8	up	0.0	norm	buff	H
62.0	fem	asy...	140.0	268.0	fal	hyp	160.0	fal	3.6	down	2.0	norm	sick	S3
57.0	fem	asy...	120.0	354.0	fal	norm	163.0	true	0.6	up	0.0	norm	buff	H
63.0	male	asy...	130.0	254.0	fal	hyp	147.0	fal	1.4	flat	1.0	rev	sick	S2
53.0	male	asy...	140.0	203.0	true	hy	155.0	true	3.1	down	0.0	rev	sick	S1

上图中很多数据都是文字描述的，在数据预处理的过程中需要根据每个字段的含义将字符转为数值。

二值类的数据
比如sex字段有female和male两种形式，可以将female表示成0，male表示成1。

多值类的数据
比如cp字段，表示胸部的疼痛感，可以将疼痛感由轻到重映射成0～3的数值。

数据预处理通过sql脚本来实现，具体请参考SQL脚本组件。

```
select age,
(case sex when 'male' then 1 else 0 end) as sex,
(case cp when 'angina' then 0 when 'notang' then 1 else 2 end) as cp,
trestbps,
chol,
(case fbs when 'true' then 1 else 0 end) as fbs,
(case restecg when 'norm' then 0 when 'abn' then 1 else 2 end) as restecg,
thalach,
(case exang when 'true' then 1 else 0 end) as exang,
oldpeak,
(case slop when 'up' then 0 when 'flat' then 1 else 2 end) as slop,
ca,
(case thal when 'norm' then 0 when 'fix' then 1 else 2 end) as thal,
(case status when 'sick' then 1 else 0 end) as ifHealth
from ${t1};
```

2. 特征工程

特征工程主要包括特征的衍生、尺度变化等功能。本案例中有两个组件负责特征工程部分。

过滤式特征选择

判断每个特征对于结果的影响，通过信息熵和基尼系数来表示。右键单击组件，选择**查看评估报告**显示最终结果，如下图所示。

归一化

将每个特征的数值范围变为0到1之间，可以去除量纲对结果的影响，公式为result=(val-min)/(max-min)。本次实验通过逻辑回归二分类来进行模型训练，需要每个特征去除量纲的影响。归一化结果如下图所示。

数据探查 - pai_temp_2954_36756_1 - (仅显示前一百条)

sex	cp	fbs	restecg	exang	slop	thal	ifhealth	age	trestbps	chol	thalach	oldpeak
1	0	1	1	0	1	0.5	0	0.70...	0.4811320...	0.244...	0.603053...	0.370967...
1	1	0	1	1	0.5	0	1	0.79...	0.6226415...	0.365...	0.282442...	0.241935...
1	1	0	1	1	0.5	1	1	0.79...	0.2452830...	0.235...	0.442748...	0.419354...
1	0.5	0	0	0	1	0	0	0.16...	0.3396226...	0.283...	0.885496...	0.564516...
0	1	0	1	0	0	0	0	0.25	0.3396226...	0.178...	0.770992...	0.225806...
1	1	0	0	0	0	0	0	0.5625	0.2452830...	0.251...	0.816793...	0.129032...
0	1	0	1	0	1	0	1	0.6875	0.4339622...	0.324...	0.679389...	0.580645...
0	1	0	0	1	0	0	0	0.58...	0.2452830...	0.520...	0.702290...	0.096774...
1	1	0	1	0	0.5	1	1	0.70...	0.3396226...	0.292...	0.580152...	0.225806...
1	1	1	1	1	1	1	1	0.5	0.4339622...	0.175...	0.641221...	0.5
1	1	0	0	0	0.5	0	0	0.58...	0.4339622...	0.150...	0.587786...	0.064516...

3. 模型训练和预测

监督学习就是已知结果来训练模型。因为已经知道每个样本是否患有心脏病，因此本次实验是监督学习。解决的问题是预测一组用户是否患有心脏病。

拆分
通过拆分组件将数据分为两部分，本次实验按照训练集和预测集7：3的比例拆分。训练集数据流入逻辑回归二分类组件用来训练模型，预测集数据进入预测组件。

逻辑回归二分类
逻辑回归是一个线性模型，通过计算结果的阈值实现分类（具体的算法详情请自行查阅资料）。逻辑回归训练好的模型可以在模型页签中查看。

逻辑回归输出

字段名	权重	
	1	0
sex	1.473569994686197	-
cp	2.730064736238172	-
fbs	-0.6007338270729394	-
restecg	0.8990240712157691	-
exang	0.9026382341453308	-
slop	1.041821068646534	-
thal	1.562393603912368	-
age	-0.4278050593226199	-

1、PAI平台提供的逻辑回归可用于多分类的，采取的策略是OneVsAll，因此在多分类的情况下，会出现多个方程，每个方程针对目标特征的某个value值，即权重（weight）下方对应的列名；

2、逻辑回归的完整公式为：$o(z) = 1 / (1 + \exp(z))$； $z = w0 + w1 * x1 + w2 * x2 + ... + wm * xm$。（其中 x1, x2, ..., xm 是某样本数据的各个特征，w1, w2, ... 是特征的权重值）

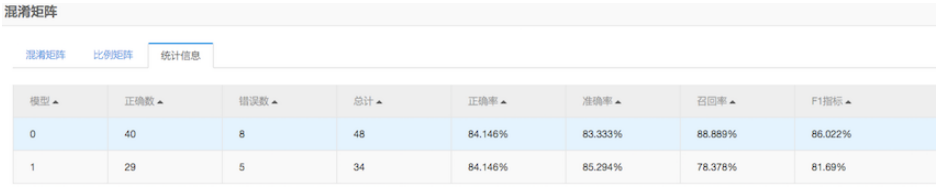
关闭

预测

预测组件的两个输入桩分别是模型和预测集。预测结果展示的是预测数据、真实数据、每组数据不同结果的概率。

4. 评估

通过混淆矩阵组件可以查看模型的准确率等参数。



通过此组件可以方便地根据预测的准确性来评估模型。

总结

通过以上数据探索流程可以得到以下结论。

模型权重

- 通过每个模型对应特征的权重，可以大体分析出特征对结果的影响大小。如果模型权重如下图

featname ▲	weight ▲
thalach	0.1656917122459715 7
oldpeak	0.1464069761877935 2
thal	0.1376916655990601 5
ca	0.1146709754621757 5
chol	0.1026770957660085 9
age	0.0787643048452784 1
trestbps	0.0772599125640569
slop	0.0770276260907830 6
restecg	0.0152468324974051 05
cp	0.0037507283721422424
exang	0
fbs	0
sex	0

- thalach（心跳数）对于是否发生心脏病影响最大。
- 性别对于是否发生心脏病没有影响。

模型效果
通过本文档提供的14个特征，心脏病预测准确率可以达到百分之八十以上。模型可以用来做预测，辅助医生预防和治疗心脏病。

新闻分类案例

本文数据为虚构，仅供实验。

本实验拟在介绍文本类组件。如果您有相关的需求，想要提高最终的效果，请联系我们。我们为您提供完整的解决方案和商业合作。

背景

新闻分类是文本挖掘领域较为常见的场景。目前很多媒体或是内容生产商对于新闻这种文本的分类常常采用人肉打标的方式，消耗了大量的人力资源。本文通过智能的文本挖掘算法对新闻文本进行分类。无需任何人肉打标，完全由机器智能化实现。

本文通过PLDA算法挖掘文章的主题，通过主题权重的聚类，实现新闻自动分类。包括了分词、词型转换、停用词过滤、主题挖掘、聚类等流程。

数据集介绍

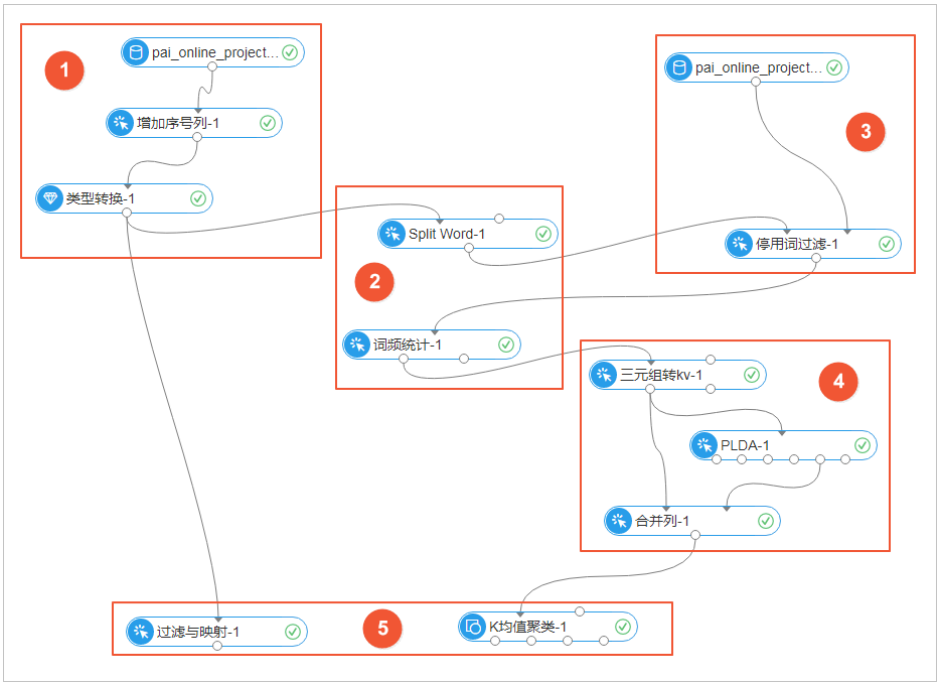
数据截图如下图所示。

具体字段如下：

字段名	含义	类型	描述
category	新闻类型	string	体育、女性、社会、军事、科技等
title	标题	string	新闻标题
content	内容	string	新闻内容

数据探索流程

实验流程图如下：



实验大致分为以下五个步骤：

- 1：增加序号列
- 2：分词及词频统计
- 3：停用词过滤
- 4：文本主题挖掘
- 5：结果分析和评估

1. 增加序号列

本实验的数据源是以单个新闻为单元，需要增加ID列来作为每篇新闻的唯一标识，方便下面算法的计算。

2. 分词及词频统计

这两步都是文本挖掘领域最常规的做法。

首先使用分词组件对**content**字段（新闻内容）进行分词。**去除过滤词**之后（过滤词一般是标点符号及助语），再对词频进行统计。结果如下图所示。

append_id ▲	word ▲	count ▲
0	分分	1
0	别墅	1
0	勇敢	1
0	包装	1
0	博爱	1
0	却	1
0	又	2
0	发	1

3. 停用词过滤

停用词过滤组件用于过滤输入的停用词词库，一般过滤标点符号以及对文章影响较小的助语等。

4. 文本主题挖掘

使用PLDA文本挖掘组件需要先将文本转换成三元形式（文本转数字），结果如下图所示。

append_id ▲	key_value ▲
213	337:1,412:1,667:3,861:1,1096:2,1582:1,1693:1,2109:1,2283:1,2371:1,2659:1,3054:3,3092:1,3232:1,4170:1,4376:1,4889:1,5206:1,5427:1,5595:1,5692:1,5739:1,6116:1,6133:1,6529:...
216	10:1,127:1,436:1,675:1,891:1,915:1,1096:2,1468:1,1757:1,2013:1,2109:1,2562:1,2783:1,3054:1,3400:1,3427:1,3443:1,3459:1,4597:1,6116:1,6183:1,6190:1,6529:1,6552:1,6871:1,7...
219	228:1,339:1,394:1,430:2,539:3,862:1,926:1,1224:1,1421:1,1488:2,1528:1,1670:2,1822:1,1909:2,2109:1,2301:1,2325:1,2411:1,2783:1,2959:1,2983:2,3209:1,4168:1,4188:1,5111:1,5...
221	10:1,18:1,200:1,387:1,412:1,436:1,450:2,472:4,555:2,563:2,637:1,639:2,667:1,813:1,856:1,913:1,1416:1,1502:1,1604:1,1636:1,2448:1,2641:2,2659:1,2929:1,3054:3,3092:2,3100:1,...
224	1582:1,3288:1,3702:1,5582:1,5932:1,6077:1,6249:1,6430:1,6529:1,6734:1,7636:1,8888:1,9418:1,9425:1,9925:1,10017:1,10176:1,11681:1,11683:1,12744:2,12748:2
227	10:1,368:1,539:1,675:1,915:1,926:1,960:1,1096:2,1423:1,1757:1,1759:1,2057:1,2109:1,2812:1,3024:1,3092:1,3181:1,3359:1,3591:1,4514:1,5464:1,6077:1,6116:1,6295:1,6529:1,65...
23	10:10,18:3,23:1,30:1,36:1,99:2,102:6,146:1,181:2,183:1,234:1,299:1,430:1,436:1,535:1,539:2,667:2,753:1,813:5,854:1,917:1,920:1,922:1,969:5,978:2,996:1,998:1,1001:4,1096:1,11...
232	12:1,13:1,18:1,69:2,146:1,200:1,234:2,329:1,370:2,565:2,571:2,605:1,608:2,667:7,813:3,891:6,1008:5,1065:1,1096:1,1104:1,1189:5,1190:2,1293:1,1572:1,1636:1,1816:1,2117:1,21...
235	12:2,13:2,18:1,88:1,204:1,478:1,523:1,558:1,575:1,606:1,667:2,670:1,754:2,803:1,872:1,921:1,1119:1,1398:2,1421:1,1498:1,1704:1,1947:1,2109:2,2132:1,2352:1,2783:3,3019:1,30...
238	10:3,202:2,539:1,667:1,892:1,1096:3,1127:1,1584:1,1806:2,2109:1,2122:1,2143:1,3024:1,3054:2,3364:1,3701:2,3765:1,3879:1,3984:1,5500:1,5685:1,6116:1,6529:1,6832:1,7460:1,...
240	10:1,107:1,115:1,146:1,412:1,430:1,450:2,596:1,667:1,800:1,931:1,1478:1,1584:1,1604:1,1652:2,1848:1,2352:1,2641:1,2676:1,2783:1,3000:2,3019:1,3054:2,3078:1,3577:1,3901:1,...

- **append_id** 是每篇新闻的唯一标识。
- **key_value** 字段中冒号前面的数字表示的是单词抽象成的数字标识，冒号后面是对应的单词出现的频率。

数据进入PLDA算法。

PLDA算法又叫主题模型，算法可以定位代表每篇文章的主题的词语。本次试验设置了50个主题，PLDA有六个输出桩，第五个输出桩输出结果展示的是每篇文章对应的每个主题的概率，如下图所

示。

docid	topic_0	topic_1	topic_2	topic_3	topic_4	topic_5	topic_6	topic_7	topic_8	topic_9	topic_10	topic_11	topi
0	0.0015625	0.0015625	0.0015625	0.0171875	0.0015625	0.0484375	0.0015625	0.0015625	0.0015625	0.0015625	0.0015625	0.0328125	0.00
1	0.001298...	0.014285...	0.001298...	0.014285...	0.001298...	0.001298...	0.014285...	0.001298...	0.001298...	0.014285...	0.1831168...	0.0012987...	0.00
2	0.011224...	0.021428...	0.001020...	0.011224...	0.011224...	0.001020...	0.001020...	0.001020...	0.001020...	0.011224...	0.0010204...	0.0214285...	0.00
3	0.000884...	0.000884...	0.000884...	0.000884...	0.000884...	0.000884...	0.000884...	0.000884...	0.000884...	0.000884...	0.0716814...	0.0008849...	0.00
4	0.039285...	0.003571...	0.003571...	0.289285...	0.003571...	0.003571...	0.003571...	0.003571...	0.003571...	0.039285...	0.0035714...	0.075	0.03
5	0.001408...	0.001408...	0.001408...	0.001408...	0.001408...	0.001408...	0.001408...	0.001408...	0.001408...	0.043661...	0.0295774...	0.0014084...	0.11
6	0.002736...	0.010199...	0.010199...	0.000248...	0.000248...	0.040049...	0.000248...	0.000248...	0.000248...	0.000248...	0.0201492...	0.0002487...	0.00
7	0.000543	0.000543	0.000543	0.000543	0.000543	0.027717	0.000543	0.000543	0.000543	0.000543	0.0548913	0.0059782	0.00

5. 结果分析和评估

上面的步骤将文章从主题的维度表示成了一个向量。

下面就可以通过向量的距离实现聚类，从而实现文章分类。K均值聚类组件的分类结果如下图所示。

docid	cluster_index
115	0
292	0
248	0
166	0
116	2
210	3
8	4
15	4

- **cluster_index** 表示的是每一类的名称。
- 找到第0类，一共有 **docid** 为115，292，248，166四篇文章。

通过过滤与映射组件查询115，292，248，166四篇文章。结果如下图所示。

append_id	category	title	content
115	体育	“欧洲通...	来源：重庆晚报“欧洲通行证”考验门将每次大赛，新推出的用球都会成为球员和市场关注的焦点，此次欧洲杯的用球“欧洲通行证”估计也会让门将们大伤脑...
166	财经	新旗舰...	机构：周四上证指数快速击穿新低进一步摧毁了市场在３０００点一带进行抵抗的信心，大盘如同自由落体，直至２９００点附近才出现抵抗，最终当天再...
248	体育	图文：...	来源：体育体育讯　北京时间６月１５号凌晨，０８欧洲杯Ｄ组第二轮开战，在奥地利因斯布鲁克的蒂沃利球场，西班牙２比１险胜瑞典，斗牛士军团以６...
292	科技	ＬＧ第...	赛迪网讯６月３０日消息，据台湾媒体报道，随着第二季度摩托罗拉在全球的手机市场的表现持续低迷，ＬＧ电子第二季度手机出货量有望突破３，０００...

实验效果并不十分理想，上图中将一篇财经、一篇科技的新闻跟两个体育类新闻分到了一起。

主要原因如下：

- 没有进行细节的调优。
- 没有对数据进行特征工程处理。
- 数据量太小。

离线调度说明

背景

本文实现的是广告CTR预测的场景。广告CTR预测是广告行业的典型应用，通过历史数据训练预测模型，对于每天的增量数据进行预测，找出广告的CTR符合标准的样本进行投放。
整套实验使用了阿里云机器学习进行数据挖掘，通过大数据开发套件进行调度和推送。具体的业务场景是：

- 通过历史数据在阿里云机器学习平台上进行模型训练。
- 通过大数据开发套件对模型进行调度。
- 每天凌晨对广告投放进行CTR预测，甄选出符合标准的广告进行推送。

数据集介绍

具体字段如下表。

字段名	含义	类型	描述
id	ID	string	广告的唯一标识
age	年龄	double	广告投放人群的年龄
sex	性别	double	广告投放人群的性别，1代表男，0代表女
duration	时长	double	广告在界面的停留时长，以秒为单位
place	位置	double	广告投放位置，0～4，按照投放位置从上到下的顺序排列
ctr	广告CTR	double	广告点击量除以展现量，大于0.03是1，其它

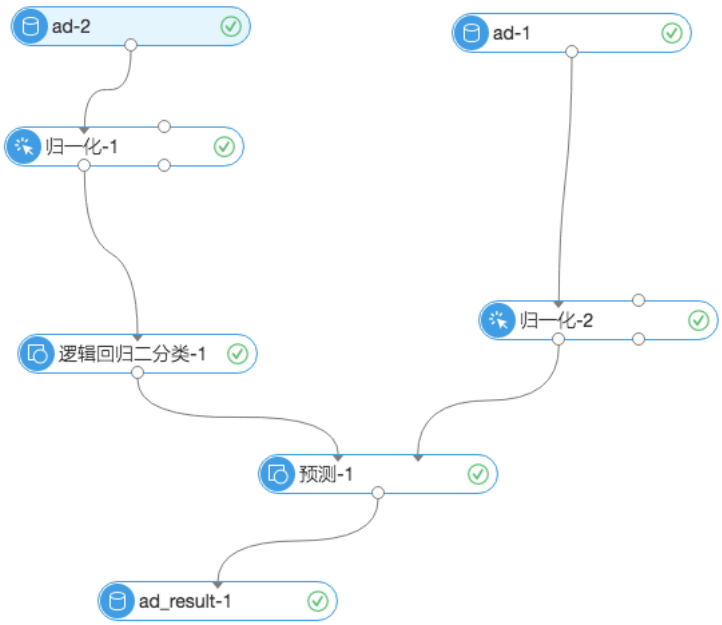
			是0
dt	partition	string	年月日，格式为yyyyMMdd

如下图所示，数据是通过random算法随机生成，所以本次实验不针对结果进行评估，主要介绍实验搭建以及和大数据开发套件的调度使用。数据包含20160919、20160920的历史数据，需要针对20160921的数据预测。使用的是MaxCompute的分区表。

id ▲	age ▲	sex ▲	duration ▲	place ▲	ctr ▲	dt ▲
0	49	1	9	0	0	20160919
1	17	1	3	1	1	20160919
2	44	0	4	0	0	20160919
3	14	1	9	1	0	20160919
4	44	1	5	4	0	20160919
5	10	1	9	3	1	20160919
6	42	1	7	3	0	20160919
7	51	1	3	1	1	20160919
8	18	0	3	3	0	20160919
9	39	0	8	4	1	20160919
10	45	1	3	2	0	20160919
11	57	0	8	2	0	20160919
12	14	0	7	2	1	20160919

实验搭建

实验流程图如下。



实验可以大致分为四个模块，数据源导入（ad），数据预处理（归一化），模型训练（逻辑回归二分类），预测（预测）。

1. 数据源导入

- “ad-2”是训练数据源。
- “ad-1”是预测数据源。
- 通过配置分区表的partition dt=@@{yyyyMMdd}，确定预测数据是每日的增量数据，如下图所示。（分区使用详情请参见 https://help.aliyun.com/document_detail/30281.html?spm=5176.doc30276.6.126.3kX7OU ）

2. 中间过程

中间过程包括数据归一化和模型训练两个步骤。模型训练是通过历史数据训练生成的预测模型。（详细原理可以参考心脏病预测案例）

3. 预测

预测生成的结果表为“ad_result-1”，数据如下图所示。

id	prediction_result	prediction_score	prediction_detail
400	0	0.5090281750932395	{"0": 0.5090281750932395, "1": 0.4909718249067604}
401	0	0.5185830406571692	{"0": 0.5185830406571692, "1": 0.4814169593428308}
402	0	0.5037390968394624	{"0": 0.5037390968394624, "1": 0.4962609031605377}
403	1	0.5136006398483877	{"0": 0.4863993601516123, "1": 0.5136006398483877}
404	0	0.5032116074286588	{"0": 0.5032116074286588, "1": 0.4967883925713412}
405	0	0.5170683273721821	{"0": 0.5170683273721821, "1": 0.4829316726278179}
406	1	0.5561919238468677	{"0": 0.4438080761531323, "1": 0.5561919238468677}
407	0	0.51090881729545	{"0": 0.51090881729545, "1": 0.48909118270455}

- prediction_result：每个广告id是否被点击。1表示被点击，0表示不被点击。
- prediction_score：对应被点击概率。

模块调度

1. 进入大数据开发套件工作空间

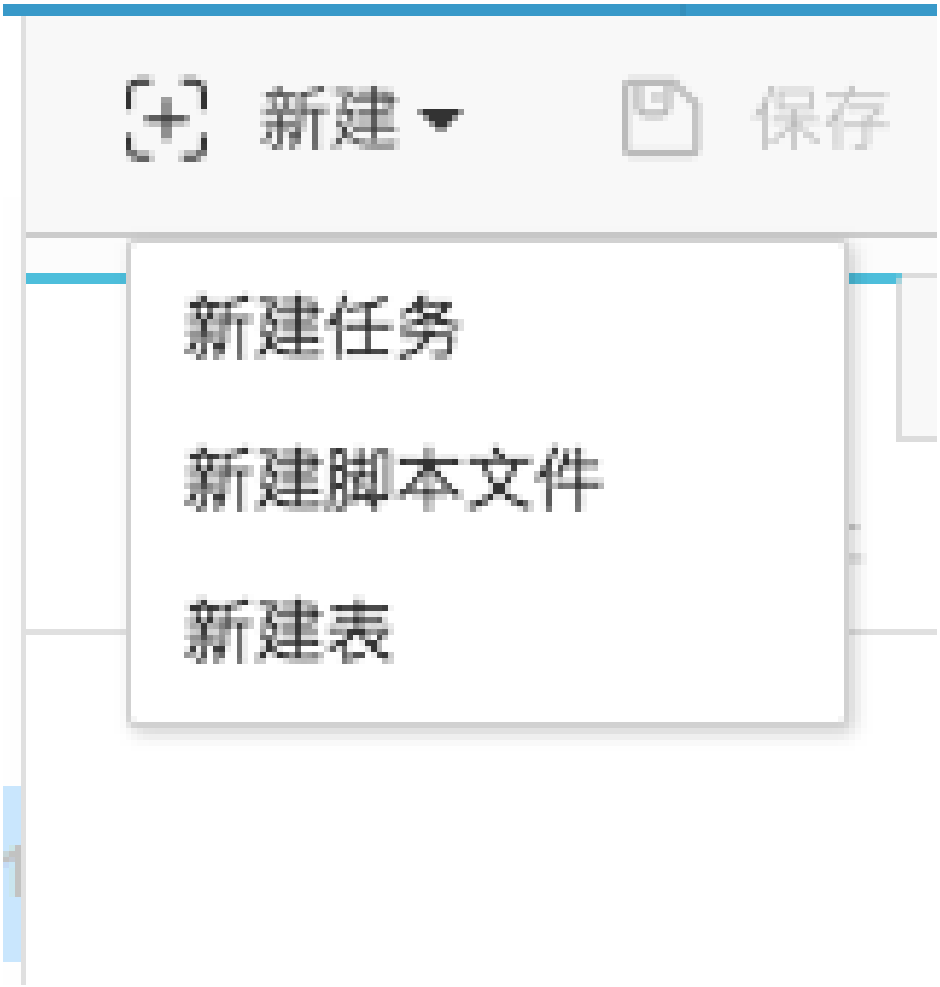
进入控制台首页，单击**DataWorks**，进入大数据开发工作空间。


大数据（数加）
数加控制台概览
DataWorks
Quick BI
机器学习
推荐引擎
公众趋势分析
DataV数据可视化
大数据计算服务

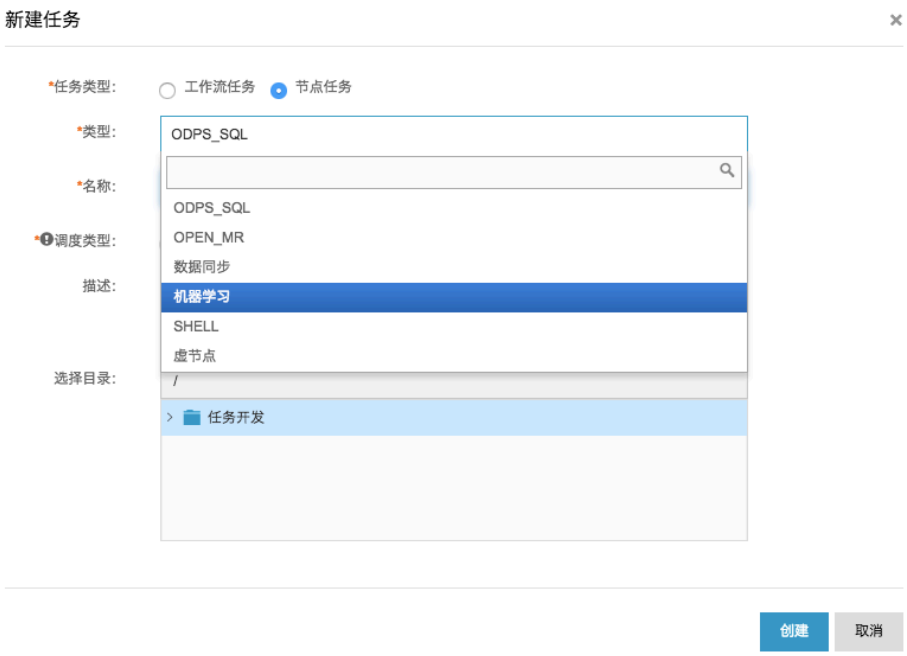
大数据开发套件与机器学习平台共用一套项目，选择需要调度的实验所在的项目，单击**进入数据开发**。

2. 新建节点调度任务

单击**新建**并选择**新建任务**。



在新建任务的配置中，**任务类型**选择**节点任务**，**类型**选择**机器学习**。



3. 配置调度任务

建立了节点任务之后，选择需要调度的机器学习实验，并在右边的配置栏选择需要调度的时间，本实验选择每日的凌晨0点进行训练和推送信息。

单击**提交**。提交的作业从**第二天**开始生效。

4. 查询任务日志

提交调度任务之后，单击**前往运维**查看日志。

人口普查统计案例

背景

本文档场景如下：
通过一份人口普查数据，根据人物的年龄、工作类型、教育程度等属性，统计学历对收入的影响。主要目的是帮助用户学习阿里云机器学习实验的搭建流程和组件的使用方式。

数据集介绍

数据源： UCI开源数据集Adult 是针对美国某区域的一次人口普查结果，共32561条数据。具体字段如下表。

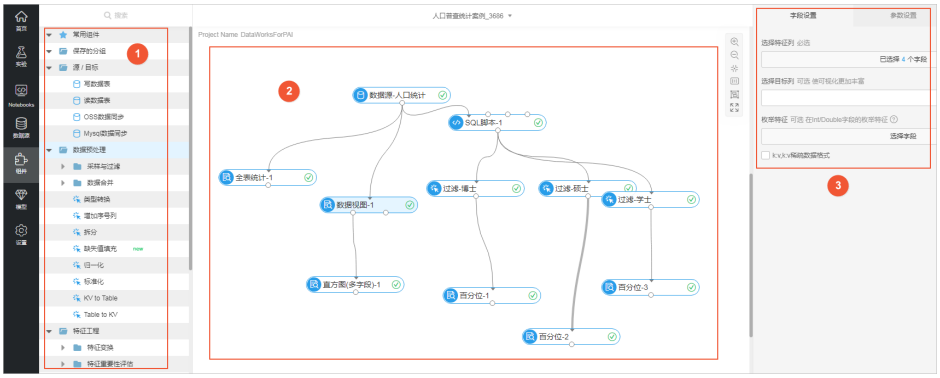
字段名	含义	类型
age	年龄	double
workclass	工作类型	string
fnlwgt	序号	string

education	教育程度	string
education_num	受教育时间	double
maritial_status	婚姻状况	string
occupation	职业	string
relationship	关系	string
race	种族	string
sex	性别	string
capital_gain	资本收益	string
capital_loss	资本损失	string
hours_per_week	每周工作小时数	double
native_country	原籍	string
income	收入	string

数据探索流程

在机器学习控制台首页，选择人口普查统计案例，单击**从模板创建**，如下图所示。

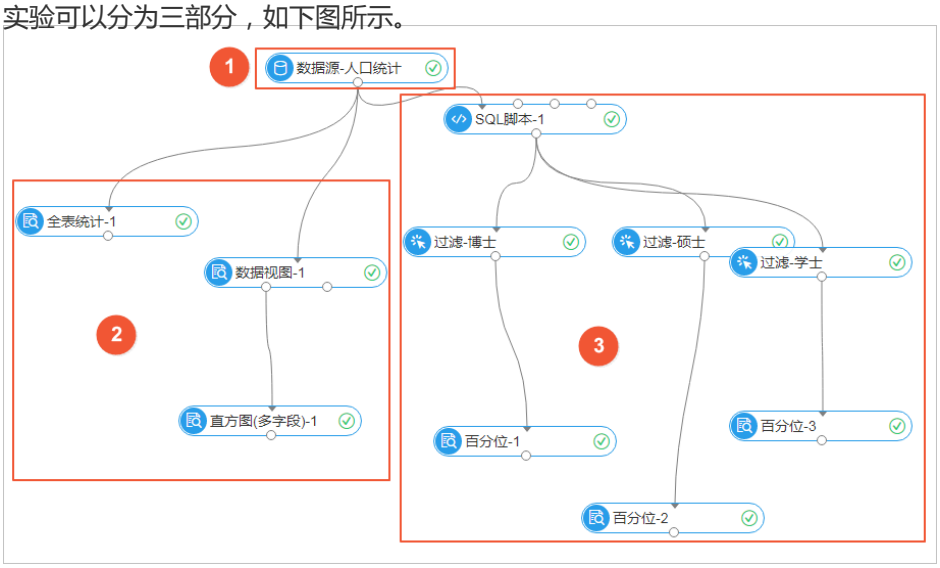
实验界面如下图所示。



- 图中第一部分为组件区域。用户可以将其拖拽到中间的空白区域搭建实验。

- 图中第二部分为实验区域。用户可以在此区域搭建实验。
- 图中第三部分为组件配置区域。用户可以在此区域配置组件参数。

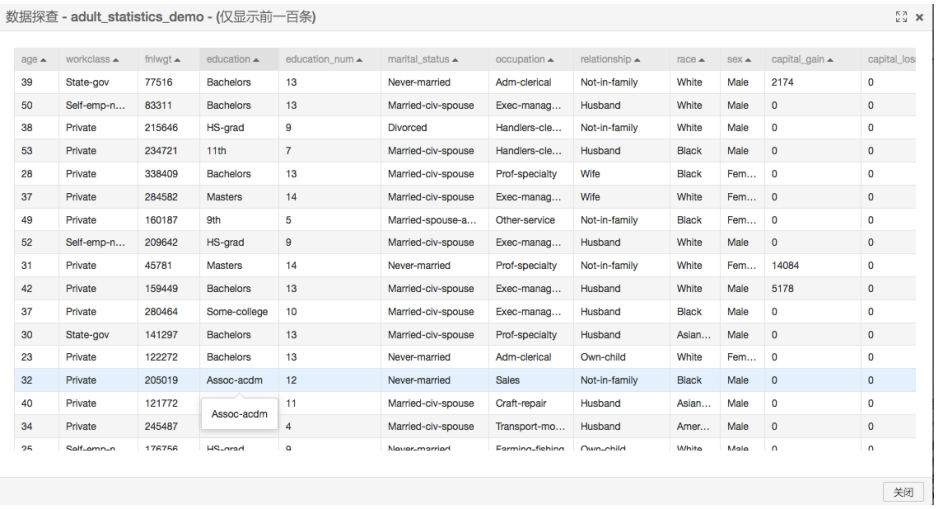
实验可以分为三部分，如下图所示。



第一部分完成数据源准备，第二部分完成数据统计，第三部分完成学历对收入的影响统计。

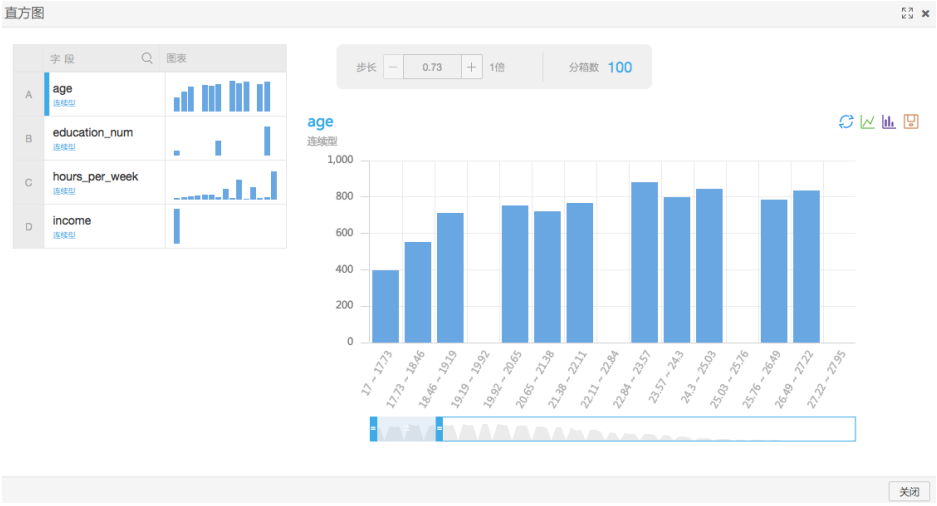
1. 数据源准备

通过机器学习IDE或者tunnel命令行工具，将数据上传到MaxCompute上。通过**读数据表**组件（图中的数据源-人口统计）读取数据。完成后右键单击组件查看数据，如下图所示。



2. 数据统计

通过全表统计和数值分布统计结果（实验中的数据视图和直方图组件）可以判断一份数据是符合泊松分布还是高斯分布、是连续还是离散。
阿里云机器学习的每个组件都提供了可视化显示结果的功能，下图是数值统计的直方图组件的输出结果，可以清楚地看到每个输入数据的分布情况。

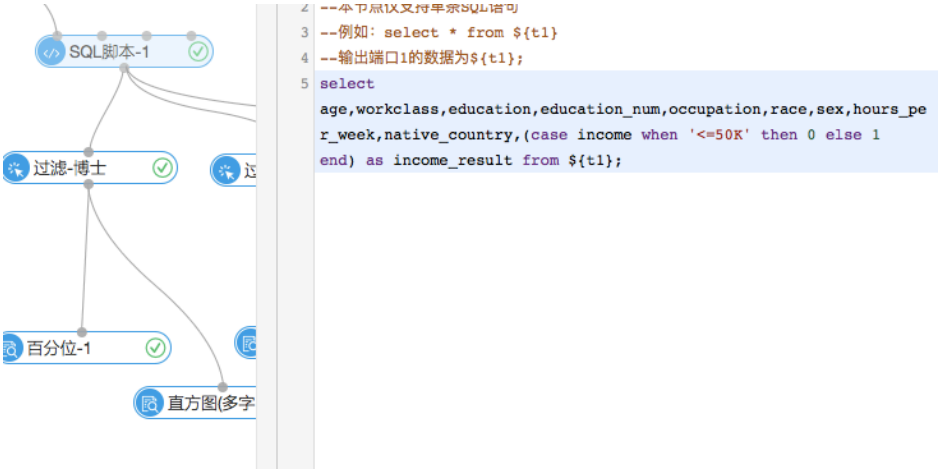


3. 学历对收入的影响统计

通过特征提取，使用机器学习算法计算得到哪些因素对收入的影响最大。本文档仅简单地针对不同学历人员的收入做统计，主要目的是介绍机器学习平台的使用方法。

数据预处理

如下图所示，数据流入的第一个组件是**SQL脚本**组件，实现数据预处理的功能。本实验是将string类型的“income”字段转换成二值型的0和1的形式。0表示年收入在50K以下，1表示年收入在50K以上（这种将文本数据数值化是机器学习特征处理的常用方式）。

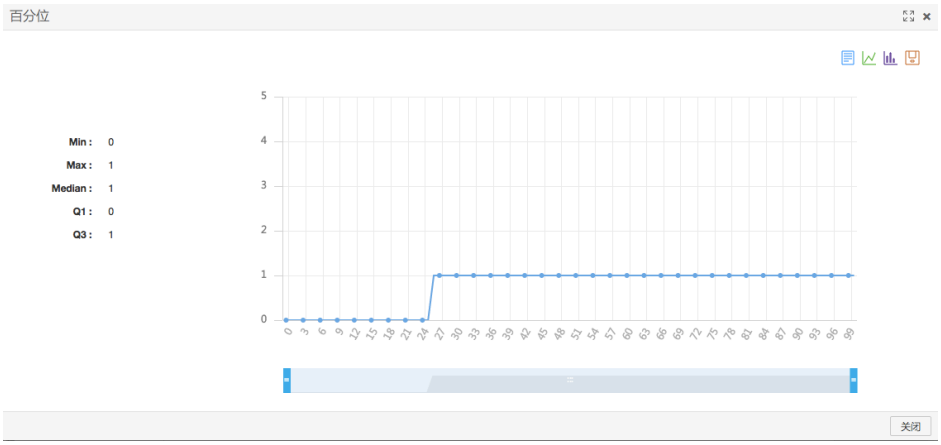


过滤与映射

通过**过滤与映射**组件将数据按照学历分为三部分，分别是博士、硕士和学士，如下图所示。
过滤与映射组件支持SQL语句，需要用户在右侧的配置栏填写where过滤条件。

结果统计

通过**百分位**组件可以得到每个分类下的收入比例。下图是折线图的展示效果，可以看到年收入在50K以下（结果中为0的点）的人群占总人数的百分之25左右。



结合三个百分位组件就可以得到如下图所示的结果。

学历	年收入大于50K的比例
博士	75%
硕士	57%
学士	42%

其它

请进入阿里云数加机器学习平台体验阿里云机器学习产品，并通过云栖社区公众号参与讨论。

学生考试成绩预测

本文数据为虚构，仅供实验。

背景

本文档通过真实的中学生数据和机器挖掘算法得到影响中学生学业的关键因素。比如父母的职业、父母的学历、家庭能否上网等。
本文档的数据采集于某中学在校生的家庭背景以及在校行为。通过逻辑回归算法生成离线模型和学业指标评估报告，对学生的期末成绩进行预测。同时生成在线预测API，通过API把训练好的离线模型应用到在线的业务场景中。

数据集介绍

数据集由25个特征列和一个目标列构成，具体字段如下表。

字段名	含义	类型	描述
sex	性别	string	F表示女，M表示男
address	住址	string	U表示城市，R表示乡村
famsize	家庭成员数	string	LE3表示少于三人，GT3表示多于三人
pstatus	是否与父母住在一起	string	T表示住在一起，A表示分开
medu	母亲的文化水平	string	从0～4逐步增高
fedu	父亲的文化水平	string	从0～4逐步增高
mjob	母亲的工作	string	分为教师相关、健康相关、服务业
fjob	父亲的工作	string	分为教师相关、健康相关、服务业
guardian	学生的监管人	string	mother、father、other
traveltime	从家到学校需要的时间	double	以分钟为单位
studytime	每周学习时间	double	以小时为单位
failures	挂科数	double	挂科次数
schoolsup	是否有额外的学习辅助	string	yes、no
fumsup	是否有家教	string	yes、no
paid	是否有相关考试学科的辅助	string	yes、no
activities	是否有课外兴趣班	string	yes、no

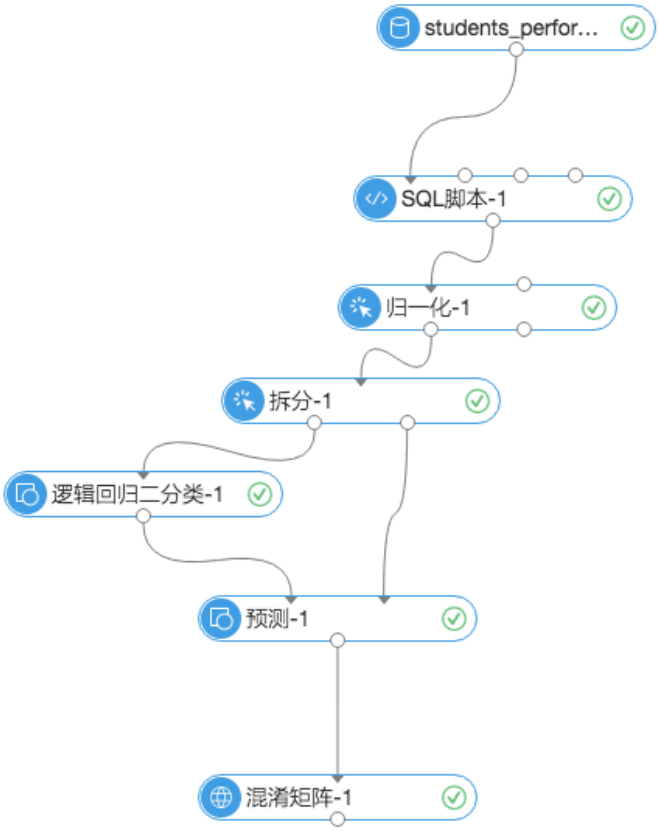
higher	是否有向上求学意愿	string	yes、no
internet	家里是否联网	string	yes、no
famrel	家庭关系	double	从1～5表示关系从差到好
freetime	课余时间量	double	从1～5从少到多
goout	跟朋友出去玩的频率	double	从1～5从少到多
dalc	日饮酒量	double	从1～5从少到多
walc	周饮酒量	double	从1～5从少到多
health	健康状况	double	从1～5表示状态从差到好
absences	出勤量	double	0到93次
g3	期末成绩	double	20分制

数据截图如下。

sex	address	famsize	pstatus	medu	fedu	mjob	fjob	guardian	traveltime	studytime	failures	schoolsup	famsup
F	U	GT3	A	4	4	at_ho...	teacher	mother	2	2	0	yes	no
F	U	GT3	T	1	1	at_ho...	other	father	1	2	0	no	yes
F	U	LE3	T	1	1	at_ho...	other	mother	1	2	3	yes	no
F	U	GT3	T	4	2	health	services	mother	1	3	0	no	yes
F	U	GT3	T	3	3	other	other	father	1	2	0	no	yes
M	U	LE3	T	4	3	services	other	mother	1	2	0	no	yes
M	U	LE3	T	2	2	other	other	mother	1	2	0	no	no
F	U	GT3	A	4	4	other	teacher	mother	2	2	0	yes	yes
M	U	LE3	A	3	2	services	other	mother	1	2	0	no	yes
M	U	GT3	T	3	4	other	other	mother	1	2	0	no	yes
F	U	GT3	T	4	4	teacher	health	mother	1	2	0	no	yes

离线训练

实验流程图如下。



数据自上到下流入实验，依次完成了数据预处理、拆分、训练、预测与评估。

1. 数据预处理

SQL脚本如下。

```
select (case sex when 'F' then 1 else 0 end) as sex,
(case address when 'U' then 1 else 0 end) as address,
(case famsize when 'LE3' then 1 else 0 end) as famsize,
(case Pstatus when 'T' then 1 else 0 end) as Pstatus,
Medu,
Fedu,
(case Mjob when 'teacher' then 1 else 0 end) as Mjob,
(case Fjob when 'teacher' then 1 else 0 end) as Fjob,
(case guardian when 'mother' then 0 when 'father' then 1 else 2 end) as guardian,
traveltime,
studytime,
failures,
(case schoolsup when 'yes' then 1 else 0 end) as schoolsup,
(case fumsup when 'yes' then 1 else 0 end) as fumsup,
(case paid when 'yes' then 1 else 0 end) as paid,
(case activities when 'yes' then 1 else 0 end) as activities,
(case higher when 'yes' then 1 else 0 end) as higher,
(case internet when 'yes' then 1 else 0 end) as internet,
famrel,
freetime,
```

```
goout,
Dalc,
Walc,
health,
absences,
(case when G3>14 then 1 else 0 end) as finalScore
from ${t1};
```

使用**SQL脚本**组件将文本数据结构化。

- 比如源数据分别有yes和no的情况，可以通过0表示yes，1表示no，将文本数据量化。
- 对于一些多种类的文本型字段，可以结合业务场景将数据抽象化。比如“Mjob”字段，是teacher表示为1，不是teacher表示为0。抽象后这个特征的意义就是表示工作是否与教育相关。
- 对于目标列，按照大于18分设为1，其它为0的方式进行量化。目的是通过训练，找出可以预测分数的模型。

2. 归一化

归一化组件的作用是去除量纲，将所有的字段都变换到0～1之间，去除字段间大小不均衡带来的影响，结果如下图所示。

sex	address	famsize	pstatus	medu	fedu	mjob	fjob	guardian	traveltime	studytime	failures	schoolsup	famsup
1	1	0	0	1	1	0	1	0	0.333333333...	0.33333333...	0	1	0
1	1	0	1	0.25	0.25	0	0	0.5	0	0.33333333...	0	0	1
1	1	1	1	0.25	0.25	0	0	0	0	0.33333333...	1	1	0
1	1	0	1	1	0.5	0	0	0	0	0.66666666...	0	0	1
1	1	0	1	0.75	0.75	0	0	0.5	0	0.33333333...	0	0	1
0	1	1	1	1	0.75	0	0	0	0	0.33333333...	0	0	1
0	1	1	1	0.5	0.5	0	0	0	0	0.33333333...	0	0	0
1	1	0	0	1	1	0	1	0	0.333333333...	0.33333333...	0	1	1
0	1	1	0	0.75	0.5	0	0	0	0	0.33333333...	0	0	1
0	1	0	1	0.75	1	0	0	0	0	0.33333333...	0	0	1
1	1	0	1	1	1	1	0	0	0	0.33333333...	0	0	1
1	1	0	1	0.5	0.25	0	0	0.5	0.666666666...	0.66666666...	0	0	1
0	1	1	1	1	1	0	0	0.5	0	0	0	0	1
0	1	0	1	1	0.75	1	0	0	0.333333333...	0.33333333...	0	0	1

3. 拆分

将数据集按照8：2的比例拆分，百分之八十用来训练模型，百分之二十用来预测。

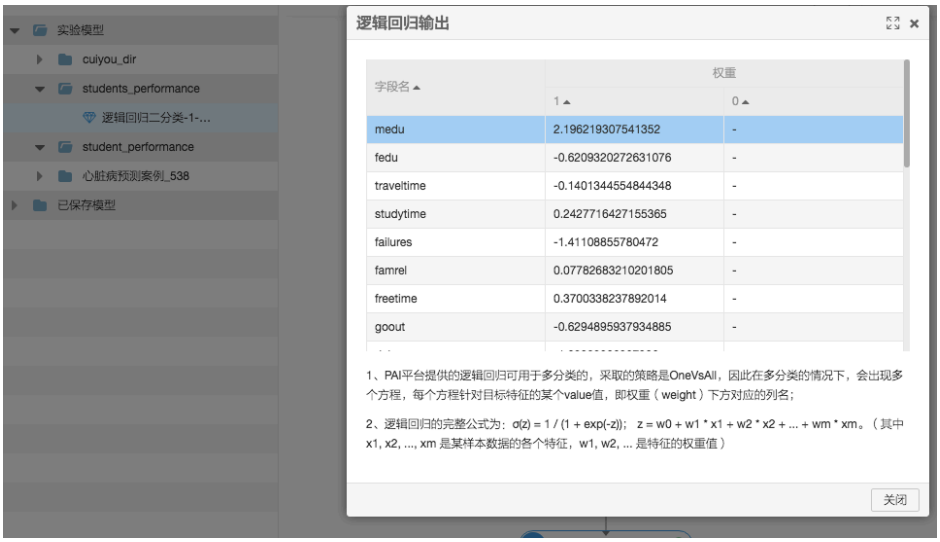
4. 逻辑回归

通过逻辑回归算法训练生成离线模型。算法详情请参见wiki。

5. 结果分析与评估

通过混淆矩阵查看模型预测的准确率。从下图中可以看到本实验的预测准确率为82.911%。

根据逻辑回归算法的特性，可以通过模型系数挖掘出一些有价值的信息。右键单击**逻辑回归二分类**组件查看模型，结果如下图所示。



根据逻辑回归算法的算法特性，权重越大表示特征对于结果的影响越大。权重为正数表示对结果1（期末高分）正相关，权重负数表示负相关。下表对几个权重较大的特征进行了分析。

字段名	含义	权重	分析
mjob	母亲的工作	-0.7998341777833717	母亲是老师对于孩子考高分是不利的。
fjob	父亲的工作	1.422595764037065	如果父亲是老师，对于孩子取得好的成绩是非常有利的。
internet	家里是否联网	1.070938672974736	家里联网不但不会影响成绩，还会促进孩子的学习。
medu	母亲的文化水平	2.196219307541352	母亲的文化水平高低对于孩子的影响是最大的，母亲文化越高孩子学习越好。

由于本次实验的数据集较小，以上分析结果不一定准确，仅供参考。

在线预测部署

生成离线模型后，可以将离线模型部署到线上，通过调用**restful-api**实现在线预测功能。详细步骤请参考**在线预测部署功能说明**。

其它

请进入**阿里云数加机器学习平台**体验阿里云机器学习产品，并通过**云栖社区公众号**参与讨论。

相似标签自动归类

背景

本文档使用机器学习平台的文本分析功能，实现一版简单的商品标签自动归类系统。具体场景如下：

双十一购物狂欢节马上又要到来了，各种关于双十一的爆品购物列表在网上层出不穷。对于经常网购的用户来说，一定清楚通常一件商品会有很多维度的标签来展示。比如一个鞋子，它的商品描述可能是“韩都少女英伦风系带马丁靴女磨砂真皮厚底休闲短靴”。如果是一个包，那么它的商品描述可能是“天天特价包包2016新款秋冬斜挎包韩版手提包流苏贝壳包女包单肩包”。

每个产品的描述都包含非常多的维度，可能是时间、产地、款式等，如何按照特定的维度将数以万计的产品进行归类，往往是电商平台最头痛的问题。其中最大的挑战是如何判断每种商品的维度由哪些标签组成。如果可以通过算法自动学习标签词语，例如“日本”、“福建”、“韩国”等与地点相关的标签，那么就可以快速地构建标签归类体系。

数据说明

数据是在网上直接下载并且整理的一份2016年双十一购物清单，一共有两千多条商品描述，每一行代表一款商品的标签聚合，如下图所示。

```
【天天特价】韩都少女英伦风系带马丁靴女磨砂真皮厚底休闲短靴
拉杆箱万向轮24寸铝框皮箱旅行箱女行李箱26寸包硬箱复古登机箱20
韩国孔孝真欧尼oni同款2016新款mini三角小包包手机包单肩斜挎包
迷失麋鹿2016秋冬独家定制款女圆头粗跟靴子沙漠靴子真皮细带短靴
MASOOMAKE秋季中跟厚底马丁靴女英伦风 2016新款系带粗跟短靴子女
羊皮毛一体雪地靴代购天天特价冬保暖运动套筒松糕防滑短筒雪地鞋
潮2016冬季新款欧美加绒高跟女靴尖头粗跟短靴马丁靴短筒棉靴女鞋
2016冬韩国内增高女棉鞋保暖加绒羊羔毛高帮系带休闲鞋韩版雪地靴
天天特价英伦学院风粗跟小皮鞋漆皮系带复古牛津鞋尖头加绒女单鞋
包包2016新款女包韩版简约复古百搭链条包单肩小包包小方包斜挎包
天天特价包包2016新款秋冬斜挎包韩版手提包流苏贝壳包女包单肩包
铝框拉杆箱万向轮行李箱男旅行箱女登机箱20/24/26寸密码箱皮箱子
艾迪猫2016秋冬新款潮单肩斜跨小包包韩版时尚女包斜挎女士小方包
天天特价欧美2016秋季新款平底单鞋懒人鞋马衔扣一脚蹬乐福鞋女鞋
韩国自制翻盖信封包圆环链条单肩包休闲气质百搭简约款女包斜挎包
```

将数据导入机器学习平台进行处理，数据上传方式请参考数据准备。

实验说明

数据上传完成后，通过拖拽机器学习组件，生成如下实验逻辑图，每一步的具体功能如下图所示。

各步骤的详细说明如下。

1. 上传数据并分词

参考**数据准备**上传shopping_data数据，代表底层数据存储。
通过**分词**组件对数据分词，分词是NLP的基础操作，本文不做介绍。

2. 增加序号列

由于上传的数据只有一个字段，需要通过增加序号列为每个数据增加主键，处理后的数据如下图所示。

content ▲	append_id ▲
【 天天 特价 】韩 都 少女 英伦风 系带 马丁 靴 女 磨砂 真皮 ...	0
拉杆箱 万向轮 24 寸 铝框 皮箱 旅行箱 女 行李箱 26 寸 包 硬...	1
韩国 孔孝真 欧尼 oni 同款 2016 新款 mini 三角 小 包包 手机...	2
迷失 麋鹿 2016 秋冬 独家 定制 款 女 圆头 粗跟靴子 沙漠 靴...	3
MASOOMAKE 秋季 中跟 厚底 马丁 靴 女 英伦风 2016 新款 ...	4
羊 皮毛一体雪地靴 代购 天天 特价 冬 保暖 运动 套筒 松糕 防...	5
潮 2016 冬季 新款 欧美 加绒 高跟女靴 尖头 粗跟短靴 马丁 靴...	6
2016 冬 韩国 内增高 女 棉鞋 保暖 加绒 羊羔毛 高帮系带休闲...	7
天天 特价 英伦 学院风 粗跟 小 皮鞋 漆皮 系带 复古 牛津 鞋 ...	8
包包 2016 新款女包 韩版 简约 复古 百搭 链条包 单肩小包包 ...	9
天天 特价包包 2016 新款 秋冬 斜挎包 韩版手提包 流苏 贝壳...	10
铝框 拉杆箱 万向轮 行李箱 男 旅行箱 女 登机箱 20 / 24 / 26 ...	11
艾迪 猫 2016 秋冬新款 潮 单肩 斜跨小包包 韩版 时尚女包 斜...	12
天天 特价 欧美 2016 秋季 新款平底单鞋 懒人 鞋 马 衔 扣 一 ...	13
韩国 自制 翻盖 信封包 圆环 链条单肩包 休闲 气质 百搭 简约...	14

3. 统计词频

展示了每个商品中出现的各种词语的个数。

4. 生成词向量

使用word2vector算法，将每个词按照意义在向量维度展开，词向量有两层含义。

- 向量距离近的两个词的真实含义比较相近，比如数据中的“新加坡”和“日本”都表示产品的产地，那么这两个词的向量距离就比较近。
- 不同词之间的距离差值也具有一定的意义，比如“北京”是“中国”的首都，“巴黎”是“法国”的首都，在训练量足够的情况下，可以得到“|中国|-|北京|=|法国|-|巴黎|”。

经过word2vector算法，将每个词被映射到百维空间上，结果如下图所示。

序号	word	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13
7	加厚	0.1177	0.009646	0.07124	-0.009802	0.008854	-0.1568	-0.2333	0.1643	0.0...	-0....	-0.2...	0.07...	-0.0...	0.02...
8	2016	0.1488	-0.1518	0.1813	0.02331	-0.03854	-0.06455	-0.001774	0.1854	0.1...	-0....	-0.2...	0.04...	0.1299	-0.0...
9	韩版	0.101	-0.02068	0.04436	0.02251	-0.1528	-0.2823	-0.2211	0.2521	0.0...	-0....	-0.2...	0.1006	0.05...	-0.0...
10	/	-0.02318	0.07028	0.189	-0.1704	0.01743	0.1096	0.1458	-0.2436	-0....	0.0...	0.1876	0.08...	-0.0...	0.1625
11	新款	0.1374	-0.05232	0.08965	0.09086	-0.09875	-0.2254	-0.1866	0.2333	0.0...	-0....	-0.189	0.07...	-0.0...	0.0253
12	6	-0.131	0.08679	0.009914	-0.3171	-0.1743	-0.1615	0.005242	-0.102	-0....	0.1...	-0.0...	0.1186	0.08...	0.1017
13	包邮	0.06004	0.04959	0.1578	0.1021	0.04368	0.1318	-0.05841	-0.01082	-0....	-0....	0.05...	0.1499	0.03...	0.0249
14	简约	-0.065	0.01107	0.02025	-0.1287	-0.09461	-0.1241	-0.05828	0.1292	-0....	0.0...	-0.0...	0.1496	0.08...	-0.1...
15	冬季	0.1803	-0.04212	0.1512	0.06145	-0.02388	-0.1422	-0.1718	0.1897	0.1...	-0....	-0.1...	0.08...	-0.0...	0.02...
16	秋冬	0.1078	-0.07883	0.1803	0.02858	-0.08247	-0.1859	-0.1708	0.2181	0.0...	-0....	-0.1...	0.1327	0.07...	-0.0...
17	-	0.04343	0.1467	0.1142	-0.2973	0.05655	0.1708	0.01833	-0.09293	-0....	0.1...	0.00...	0.1291	-0.0...	0.1162
18	纯棉	0.06417	-0.08088	0.07554	0.04668	-0.07626	-0.2355	-0.1062	0.1727	0.0...	-0....	-0.1...	0.08...	0.09...	-0.0...
19	韩国	0.0284	-0.03408	0.1062	-0.02404	-0.04606	-0.0249	-0.01154	0.05106	0.0...	0.0...	-0.0...	0.06...	0.1056	0.00...
20	家用	0.09303	0.004674	0.151	-0.08795	0.03799	0.1286	0.1244	-0.1209	0.0...	0.1...	0.07...	0.1694	0.2598	0.1493
21	g	0.06279	0.01393	0.2534	-0.01994	0.03998	0.3231	0.07817	-0.07714	-0....	0.1...	0.06...	0.1422	0.1651	0.03...
22	男	0.06893	0.009893	0.1051	0.0005736	-0.02107	-0.1202	-0.1323	0.1462	-0....	-0....	-0.1...	0.09...	-0.0...	0.0156

5. 词向量聚类

使用kmeans算法，在已经产生的词向量的基础上，计算出哪些词的向量距离比较近，并按照意义将标签词自动归类。结果展示的是每个词属于哪个聚类簇，如下图所示。

word	cluster_index
家用	83
g	83
男	79
套装	94
保暖	98
加绒	98
儿童	79
潮	90
正品	87

结果验证

通过SQL组件，在聚类簇中随意挑选一个类别，判断是否将同一类别的标签进行了自动归类，本实验选用第

10组聚类簇。

```
6 select * from ${t1} where
  cluster_index=10
```

结果如下图所示。

word ▲	cluster_index ▲
日本进口	10
俄罗斯	10
雨	10
坚果	10
台湾	10
韩国进口	10
男士内裤	10
记	10
云南	10
螺	10
油	10
新疆特产	10

通过结果中的“日本”、“俄罗斯”、“韩国”、“云南”、“新疆”、“台湾”等词可以发现系统自动将一些跟地理相关的标签进行了归类，但是里面混入了“男士内裤”、“坚果”等明显与类别不符合的标签。可能是训练样本数量不足造成的，如果训练样本足够大，那么标签聚类结果会非常准确。

TensorFlow实现图像分类

一、背景

随着互联网的发展，产生了大量的图片以及语音数据，如何对这部分非结构化数据行之有效的利用起来，一直是困扰数据挖掘工程师的一到难题。首先，解决非结构化数据常常要使用深度学习算法，上手门槛高。其次，对于这部分数据的处理，往往需要依赖GPU计算引擎，计算资源代价大。本文将介绍一种利用深度学习实现的图片识别案例，这种功能可以服用到图片的检黄、人脸识别、物体检测等各个领域。

下面尝试通过阿里云机器学习平台产品，利用深度学习框架Tensorflow，快速的搭架图像识别的预测模型，整个流程只需要半小时，就可以实现对下面这幅图片的识别，系统会返回结果“鸟”：

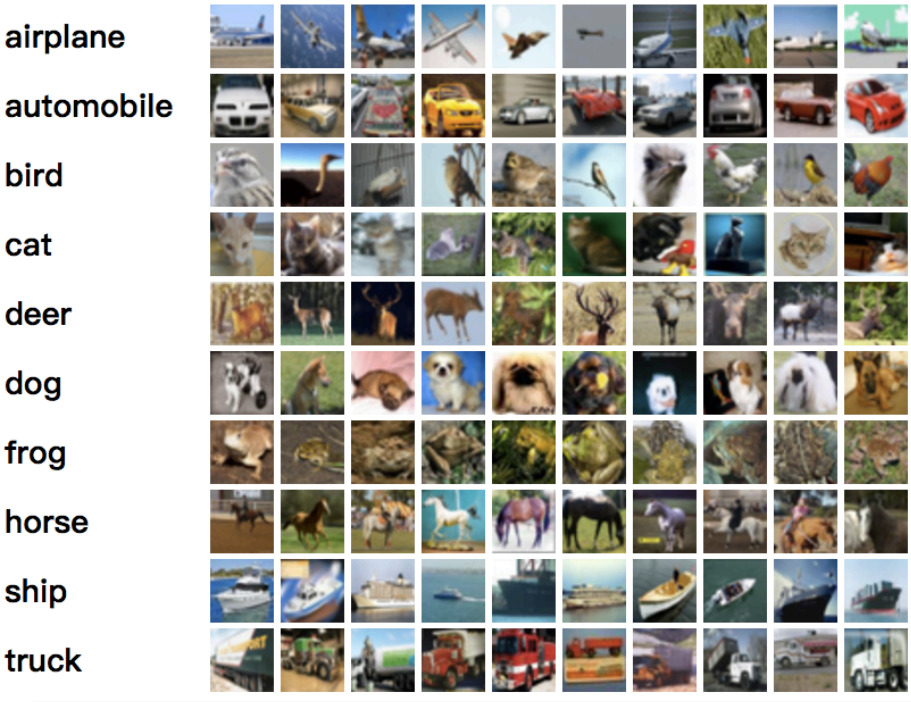
本实验能从PAI模板创建：

从模板创建的实验用户只要替换上下游两个Tensorflow组件中的checkpoint目录为自己的oss目录的可用路径即可跑通，如下图：

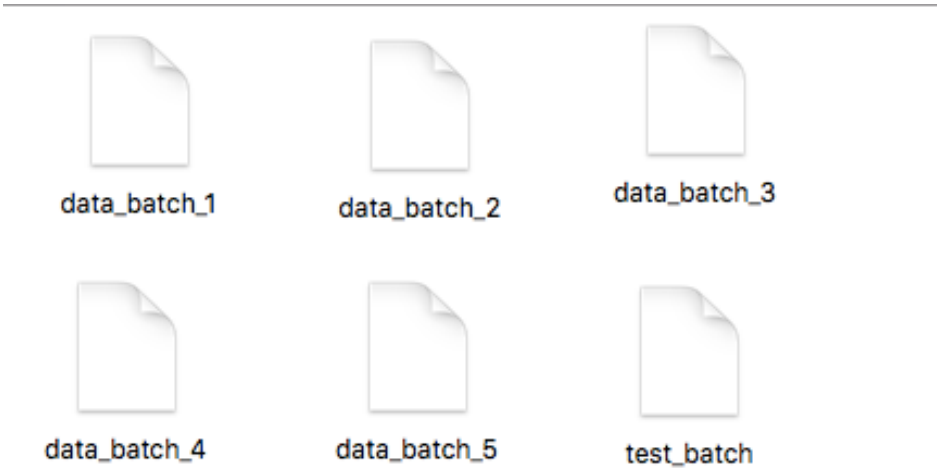
二、数据集介绍

本案例数据集及相关代码下载地址：
https://help.aliyun.com/document_detail/51800.html?spm=5176.doc50654.6.564.mS4bn9

使用CIFAR-10数据集，这份数据是一份对包含6万张像素为32*32的彩色图片，这6万张图片被分成10个类别，分别是飞机、汽车、鸟、毛、鹿、狗、青蛙、马、船、卡车。数据集截图：



数据源在使用过程中被拆分成两个部分，其中5万张用于训练，1万张用于测试。其中5万张训练数据又被拆分成5个data_batch，1万张测试数据组成test_batch。最终数据源如图：

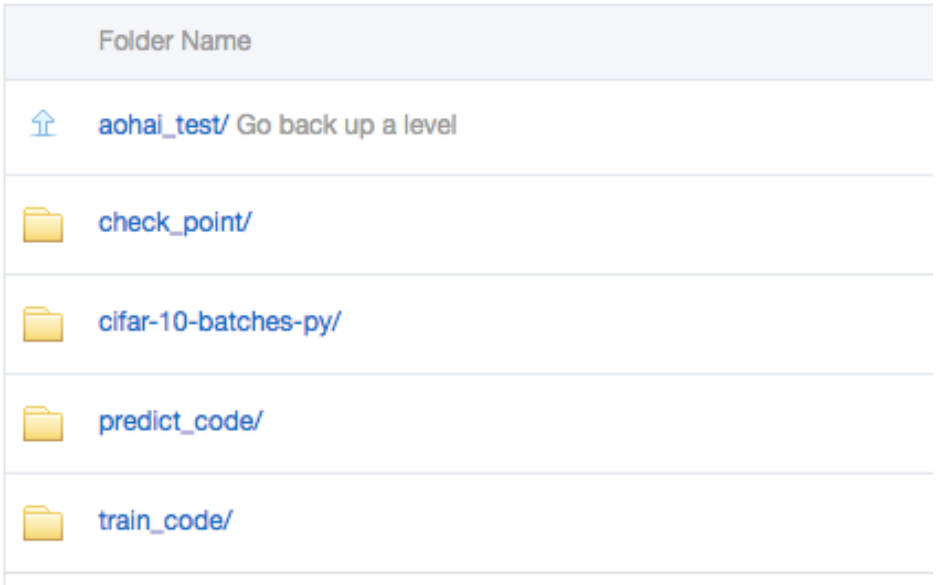


三、数据探索流程

下面我们一步一步讲解下如何将实验在阿里云机器学习平台跑通，首先需要开通阿里云机器学习产品的GPU使用权限，并且开通OSS，用于存储数据。机器学习：https://data.aliyun.com/product/learn?spm=a21gt.99266.416540.112.IOG7OUOSS：https://www.aliyun.com/product/oss?spm=a2c0j.103967.416540.50.KkZyBu

1.数据源准备

第一步，进入OSS对象存储，将本案例使用的相关数据和代码放到OSS的bucket路径下。首先建立OSS的bucket，然后我建立了aohai_test文件夹，并在这个目录下建立如下4个文件夹目录：



每个文件夹的作用如下：

check_point:用来存放实验生成的模型

cifar-10-batches-py：用来存放训练数据以及预测集数据，对应的是下载下来的数据源cifar-10-batcher-py文件和预测集bird_mount_bluebird.jpg文件

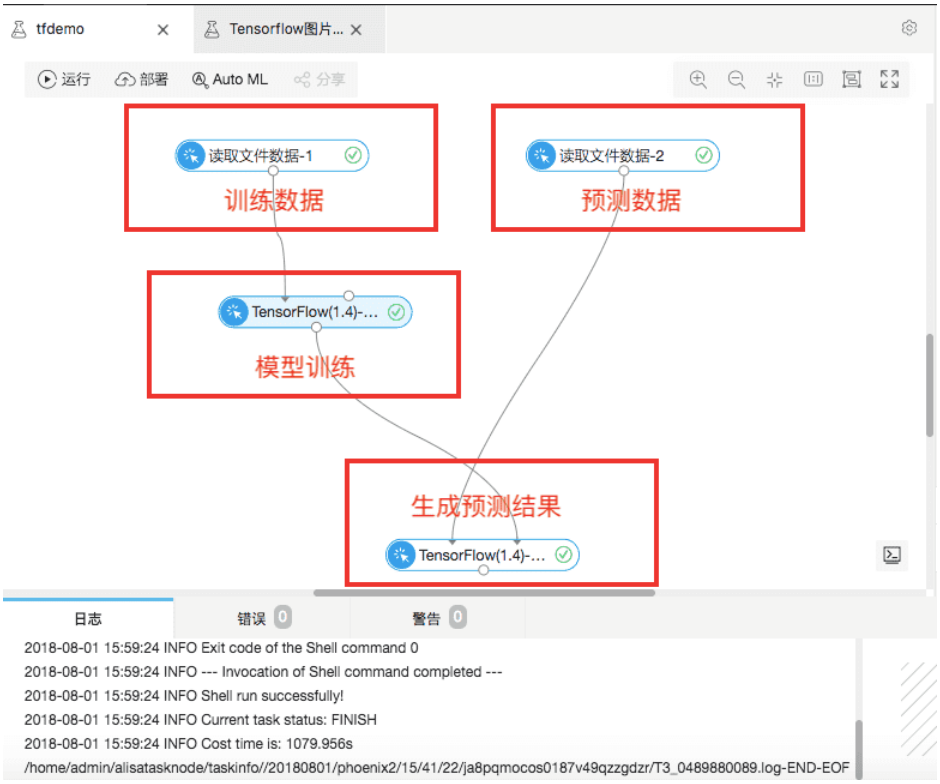
- train_code:用来存放训练数据，也就是cifar_pai.py
- predict_code:用来存放cifar_predict_pai.py

本案例数据集及相关代码下载地址：
https://help.aliyun.com/document_detail/51800.html?spm=5176.doc50654.6.564.mS4bn9

2.配置OSS访问授权

现在我们已经把数据和训练需要的代码放入OSS，下面要配置机器学习对OSS的访问，进入阿里云机器学习，在“设置”按钮的弹出页面，配置OSS的访问授权。如图：

3.搭建训练逻辑



4.训练/预测数据配置

配置训练数据和预测数据对应的OSS路径

4.模型训练

- Python代码文件：OSS中的cifar_pai.py
- 数据源目录：OSS中的cifar-10-batches-py文件夹，会自动从上游的“读文件数据“节点同步
- checkpoint输出目录/模型输入目录：OSS中的check_point文件夹，用来输出模型
- 执行调优：用来配置多机多卡相关数据

4.1代码解析

这里针对cifar_pai.py文件中的关键代码讲解：（1）构建CNN图片训练模型

```
network = input_data(shape=[None, 32, 32, 3],
```

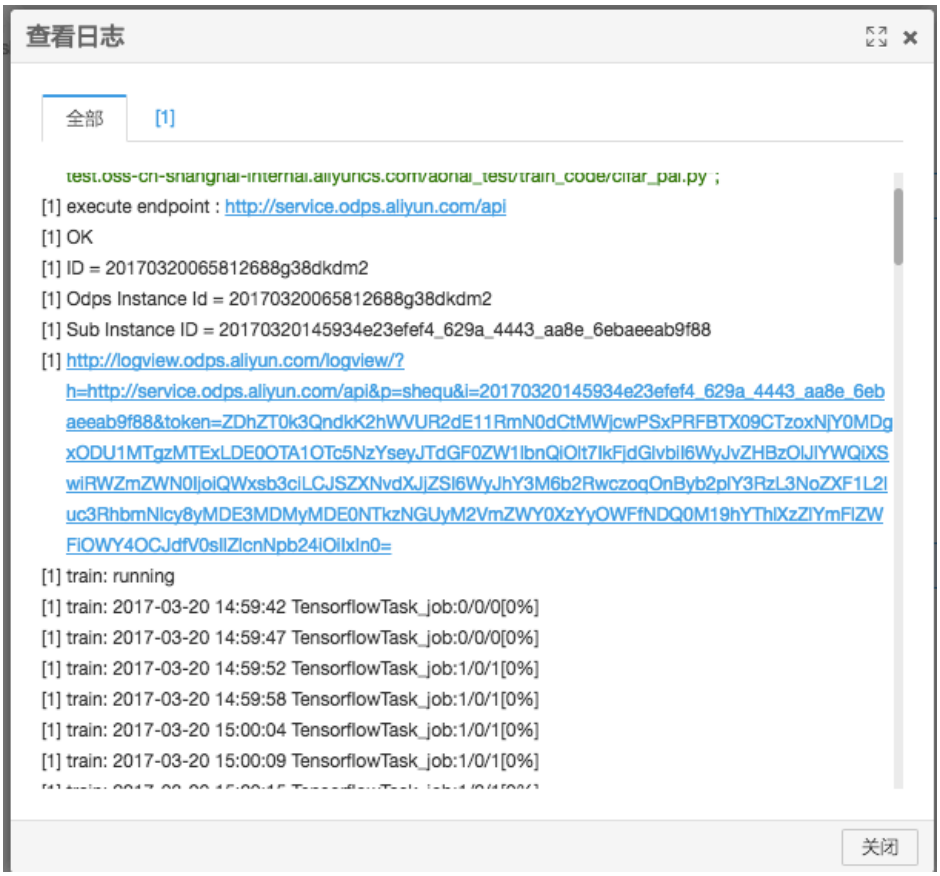
```
data_preprocessing=img_prep,
data_augmentation=img_aug)
network = conv_2d(network, 32, 3, activation='relu')
network = max_pool_2d(network, 2)
network = conv_2d(network, 64, 3, activation='relu')
network = conv_2d(network, 64, 3, activation='relu')
network = max_pool_2d(network, 2)
network = fully_connected(network, 512, activation='relu')
network = dropout(network, 0.5)
network = fully_connected(network, 10, activation='softmax')
network = regression(network, optimizer='adam',
loss='categorical_crossentropy',
learning_rate=0.001)
```

（2）训练生成模型model.tfl

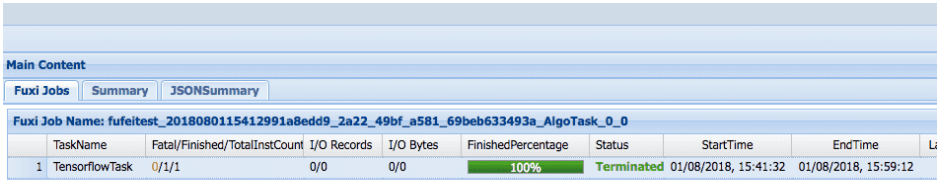
```
    model = tflearn.DNN(network, tensorboard_verbose=0)
model.fit(X, Y, n_epoch=100, shuffle=True, validation_set=(X_test, Y_test),
show_metric=True, batch_size=96, run_id='cifar10_cnn')
model_path = os.path.join(FLAGS.checkpointDir, "model.tfl")
print(model_path)
model.save(model_path)
```

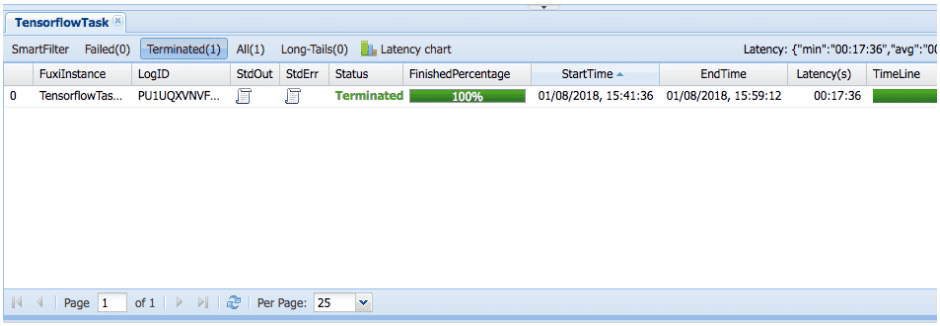
4.2代码解析

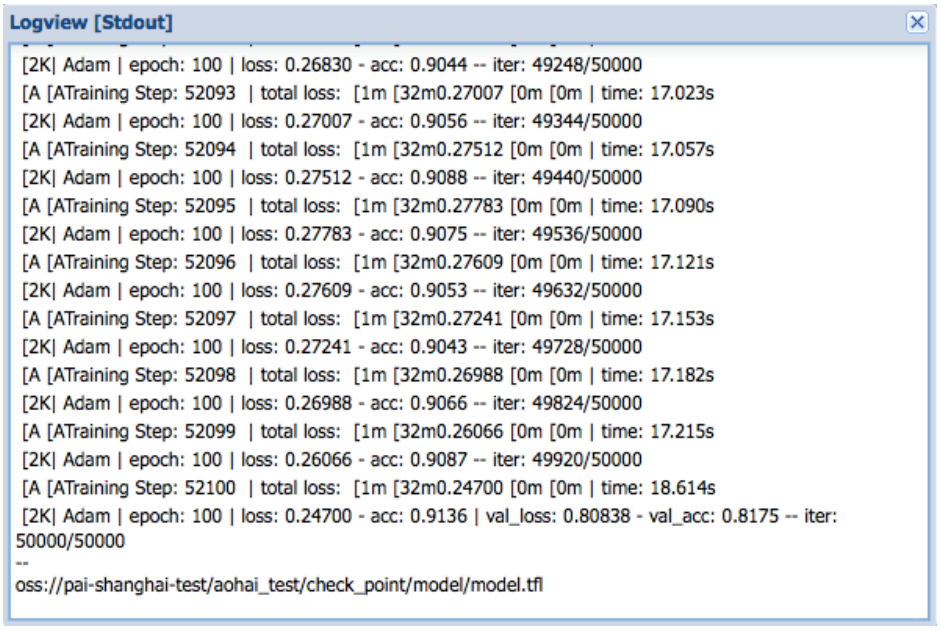
训练过程中，右键“Tensorflow”组件，点击查看日志。



点击打开logview连接，按照如下链路操作，双击打开ODPS Tasks下面的Algo Task，双击Tensorflow Task，点击Terminated，点击StdOut，可以看到模型训练的日志被实时的打印出来：



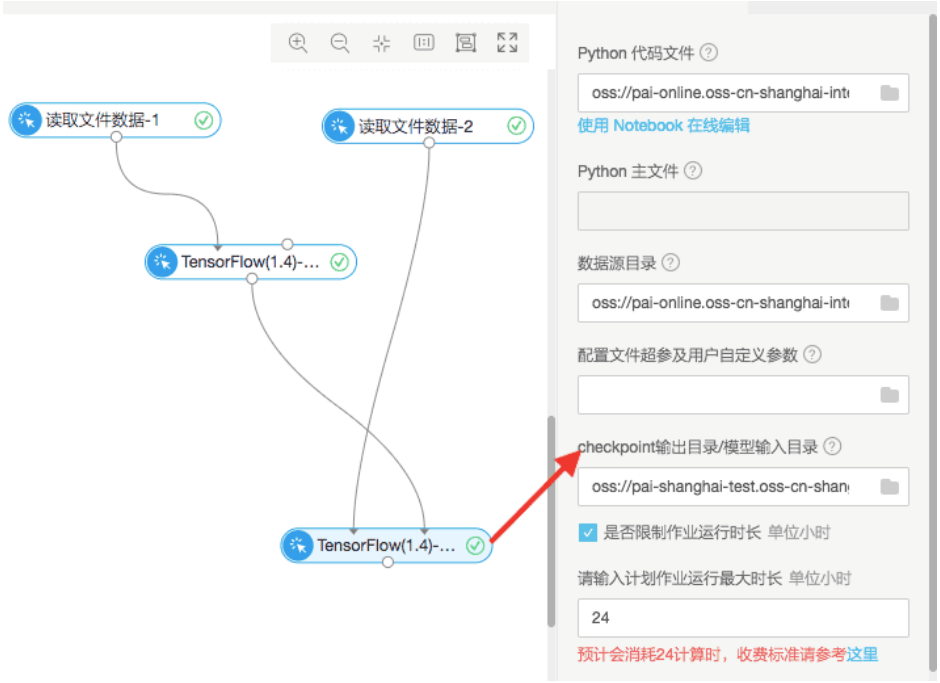




随着实验的进行，会不断打出日志出来，对于关键的信息也可以利用print函数在代码中打印，结果会显示在这里。在本案例中，可以通过acc查看模型训练的准确度。

5.结果预测

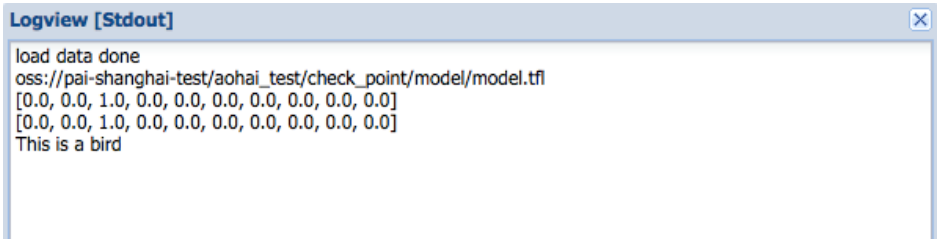
再拖拽一个“Tensorflow”组件用于预测，



- Python代码文件：OSS中的cifar_predict_pai.py
- 数据源目录：OSS中的cifar-10-batches-py文件夹,用来读取bird_mount_bluebird.jpg文件，从读文件数据组件自动同步
- checkpoint输出目录/模型输入目录：需要跟tensorflow训练组间的模型输出目录一致

预测的图片是存储在checkpoint文件夹下的图:

结果见日志，结果查看方式同步骤4：



5.1预测代码数据

部分预测代码解析：

```
  predict_pic = os.path.join(FLAGS.buckets, "bird_bullocks_oriole.jpg")
img_obj = file_io.read_file_to_string(predict_pic)
file_io.write_string_to_file("bird_bullocks_oriole.jpg", img_obj)

img = scipy.ndimage.imread("bird_bullocks_oriole.jpg", mode="RGB")

# Scale it to 32x32
img = scipy.misc.imresize(img, (32, 32), interp="bicubic").astype(np.float32, casting='unsafe')

# Predict
prediction = model.predict([img])
print (prediction[0])
print (prediction[0])
#print (prediction[0].index(max(prediction[0])))
num=['airplane','automobile','bird','cat','deer','dog','frog','horse','ship','truck']
print ("This is a %s"%(num[prediction[0].index(max(prediction[0]))]))
```

首先读入图片“bird_bullocks_oriole.jpg” ，将图片调整为像素32*32的大小，然后带入model.predict预测函数评分，最终会返回这张图片对应的十种分类
[‘airplane’,’automobile’,’bird’,’cat’,’deer’,’dog’,’frog’,’horse’,’ship’,’truck’]
的权重，选择权重最高的一项作为预测结果返回。

雾霾天气预测

背景

如果要人们评选当今最受关注话题的top10榜单，雾霾一定能够入选。如今走在北京街头，随处可见带着厚厚口罩的人在埋头前行，雾霾天气不光影响了人们的出行和娱乐，对于人们的健康也有很大危害。本文通过分析北京一年来的真实天气数据，挖掘出二氧化氮是跟雾霾天气（指PM2.5）相关性最强的污染物，从而为您揭秘形成雾霾的罪魁祸首。

登录阿里云机器学习平台，通过模板创建雾霾天气预测实验。

数据集介绍

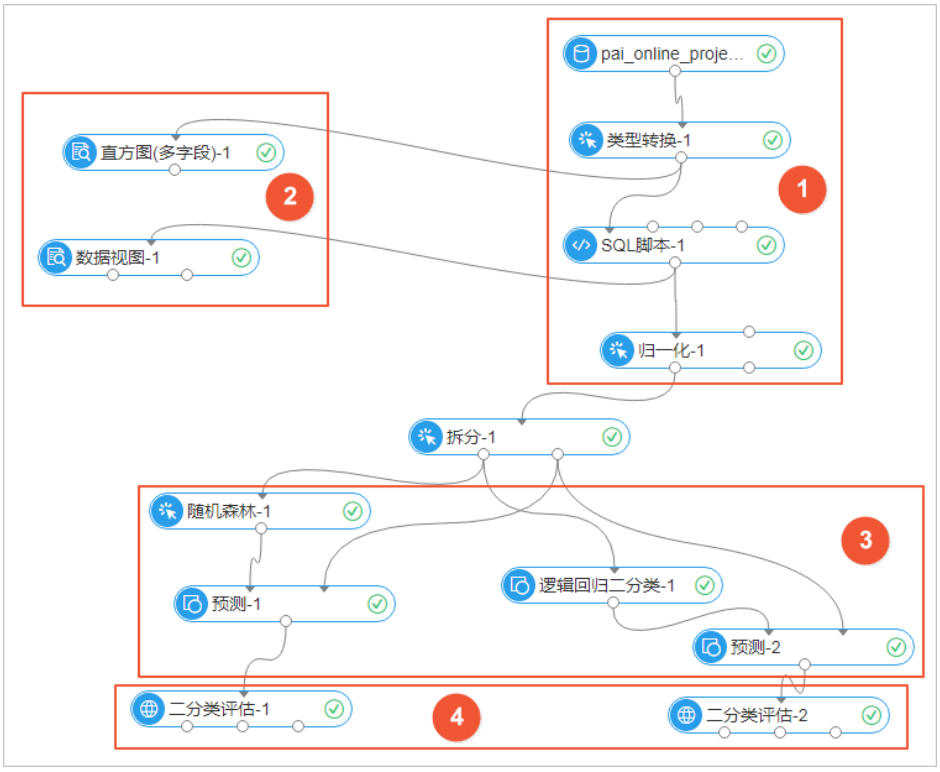
数据源：2016全年的北京天气指标。

采集的是从2016年1月1号以来每个小时的空气指标数据，具体字段如下表。

字段名	含义	类型
time	日期，精确到天	string
hour	表示的是时间，第几小时的数据	string
pm2	pm2.5的指标	string
pm10	pm10的指标	string
so2	二氧化硫的指标	string
co	一氧化碳的指标	string
no2	二氧化氮的指标	string

数据探索流程

实验流程如下。

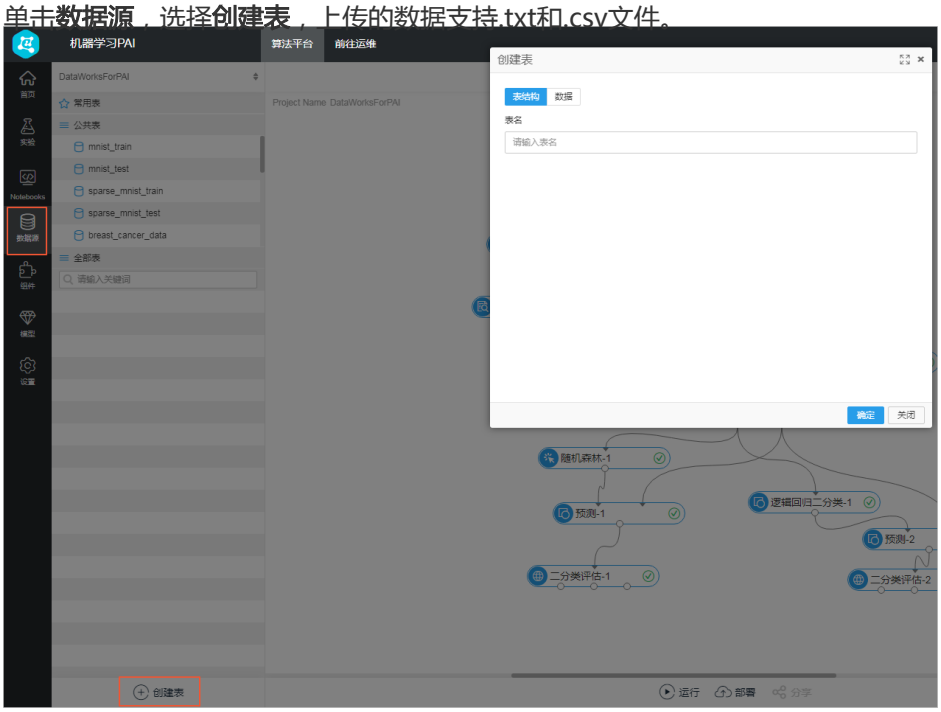


整个实验分为四部分，分别是数据导入及预处理（上图的1）、统计分析（上图的2）、模型训练及预测（上图的3）、模型评估分析（上图的4），详细介绍如下。

1. 数据导入及预处理

数据导入

单击**数据源**，选择**创建表**，上传的数据支持.txt和.csv文件。



数据导入后，右键单击组件，选择**查看数据**，结果如下。

time	hour	pm2	pm10	so2	co	no2
2016...	2	85	123	18	1.8	72
2016...	8	114	127	25	2.3	81
2016...	11	123	140	27	2.5	83
2016...	14	134	150	30	2.6	86
2016...	17	150	168	32	2.8	92
2016...	20	166	191	34	3	97
2016...	23	179	207	35	3.2	101
2016...	1	190	222	37	3.4	104
2016...	10	225	249	39	3.8	107
2016...	19	244	287	41	4	113

数据预处理

通过“类型转换”组件把string类型的数据转换成double类型。

通过“SQL脚本”组件，将目标列转换成0和1的二值类型。本实验中“pm2”列为目标列，数值超过200的作为重度雾霾天气打标为1，低于200为0，实现的SQL语句如下。

```
select time,hour,(case when pm2>200 then 1 else 0 end),pm10,so2,co,no2 from ${t1};
```

归一化

归一化的主要作用是去除量纲，即把不同指标的污染物的单位进行统一。

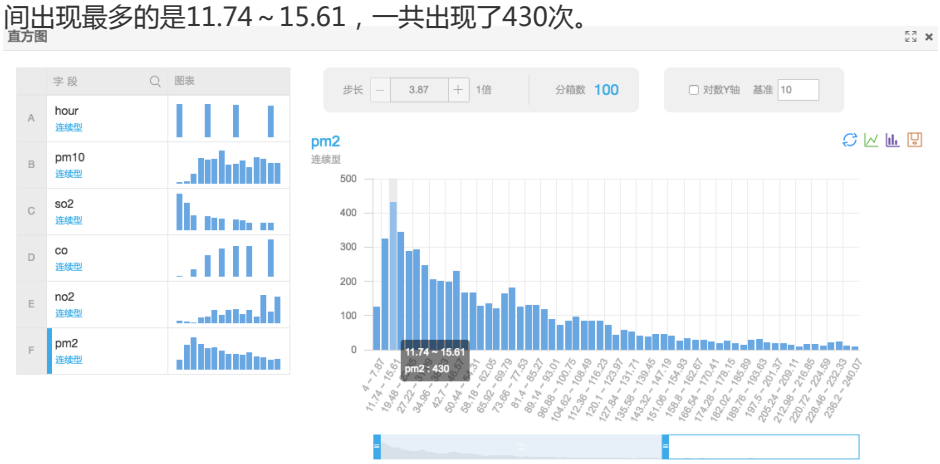
time	hour	_c2	pm10	so2	co	no2
20160101	2	0	0.24532224...	0.21917808219...	0.36956521739130427	0.43312101910828027
20160101	8	0	0.25363825...	0.31506849315...	0.4782608695652173	0.49044585987261147
20160101	11	0	0.28066528...	0.34246575342...	0.5217391304347825	0.5031847133757962
20160101	14	0	0.30145530...	0.38356164383...	0.5434782608695652	0.5222929936305732
20160101	17	0	0.33887733...	0.41095890410...	0.5869565217391303	0.5605095541401274
20160101	20	0	0.38669438...	0.43835616438...	0.6304347826086956	0.5923566878980892
20160101	23	0	0.41995841...	0.45205479452...	0.6739130434782609	0.6178343949044586
20160102	1	0	0.45114345...	0.47945205479...	0.7173913043478259	0.6369426751592356
20160102	10	1	0.50727650...	0.50684931506...	0.8043478260869563	0.6560509554140127
20160102	19	1	0.58627858...	0.53424657534...	0.8478260869565216	0.6942675159235668
20160102	22	1	0.68191268...	0.53424657534...	0.8913043478260869	0.7197452229299363
20160103	0	1	0.74428274...	0.53424657534...	0.8913043478260869	0.732484076433121
20160105	16	0	0.06860706...	0.02739726027...	0.06521739130434782	0.16560509554140126

2. 统计分析

直方图

通过“直方图”组件可以可视化地查看不同数据在不同区间下的分布。

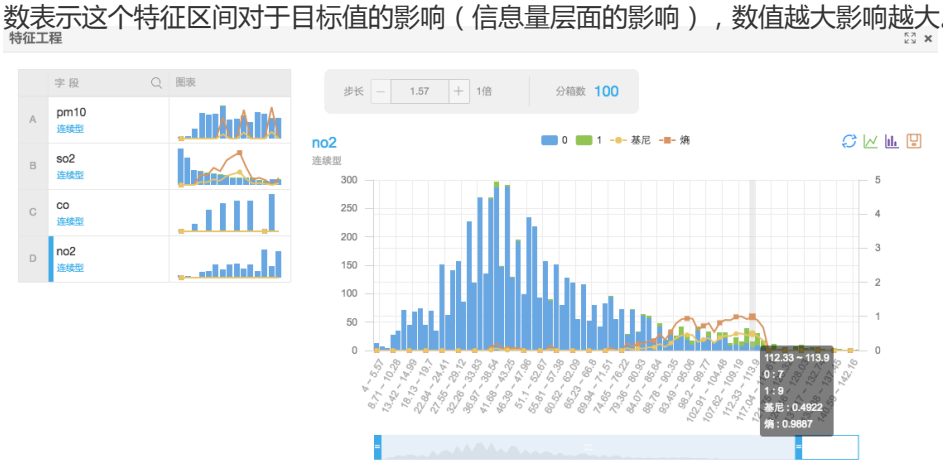
本实验通过可视化的展现，直观地看到了每个字段数据的分布情况。如下图，以PM2.5为例，数值区间出现最多的是11.74～15.61，一共出现了430次。



数据视图

通过数据视图可以查看不同指标的不同区间对于结果的影响。

如下图，以no2为例，在112.33～113.9区间产生了7个目标列为0的目标，产生了9个目标列为1的目标。即当no2在112.33～113.9区间的情况下，出现重度雾霾的天气的概率是非常大的。熵和基尼系数表示这个特征区间对于目标值的影响（信息量层面的影响），数值越大影响越大。

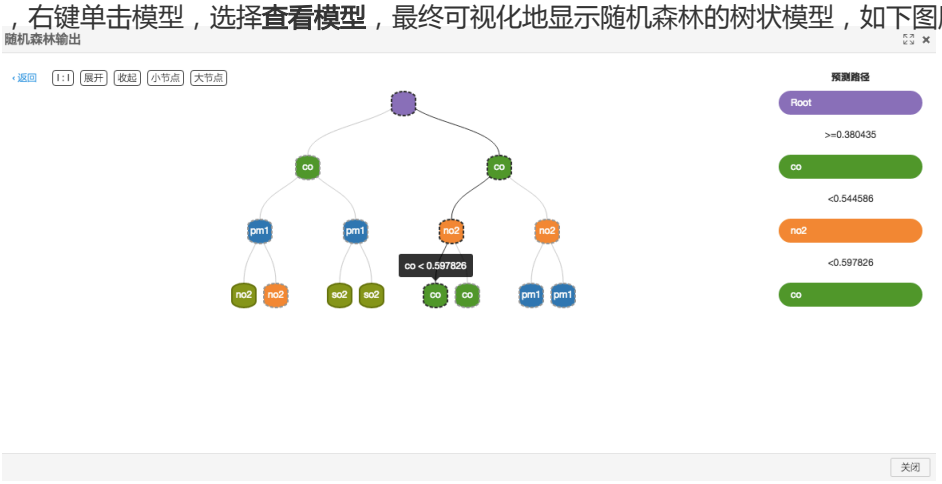


3. 模型训练及预测

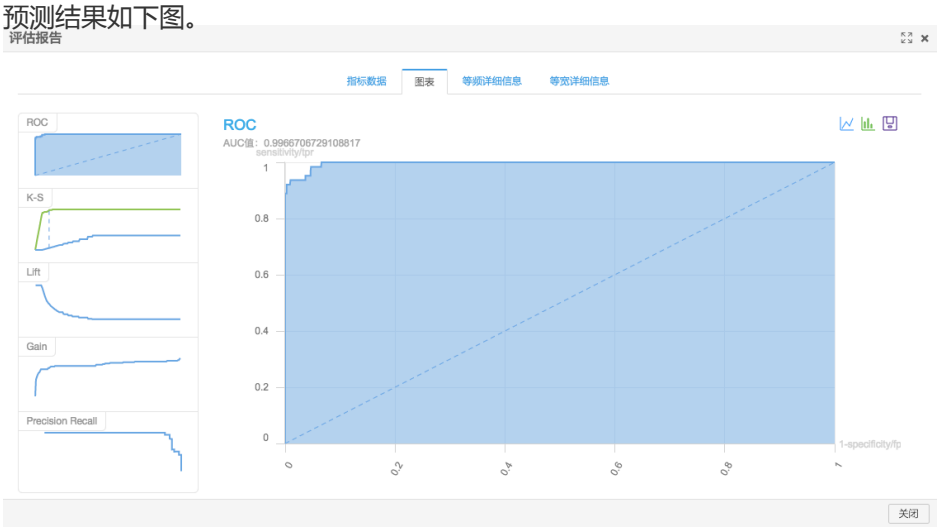
本案例采用了两种不同的算法对结果进行预测和分析，分别是随机森林和逻辑回归。

随机森林

将数据集拆分，百分之八十用来训练模型，百分之二十用来预测。单击控制台左边的**模型**，选择**已保存模型**，右键单击模型，选择**查看模型**，最终可视化地显示随机森林的树状模型，如下图所示。



预测结果如下图。



上图中的**AUC**为0.99，说明当有了本文档用到的天气指标数据，就可以预测天气是否雾霾，而且准确率可以达到百分之九十以上。

逻辑回归

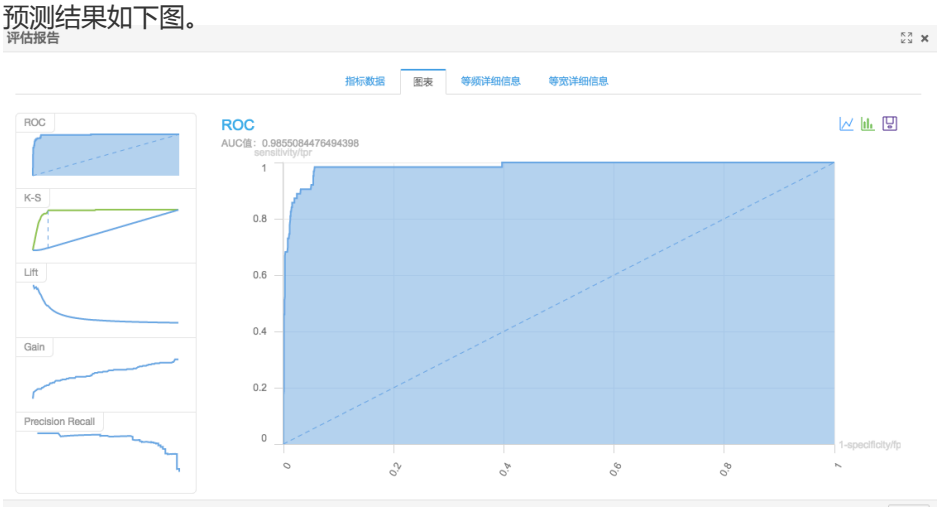
使用逻辑回归算法训练得到的是一个线性模型，如下图所示。

逻辑回归二分类

在输入数据为稀疏的时候，不显示 weight 全是 0 的特征

字段名	权重	
	1	0
pm10	18.32146628653672	-
so2	1.767062094833547	-
co	-0.2519492790928399	-
no2	10.95221282178011	-
常量	-16.66654139199668	0

预测结果如下图。

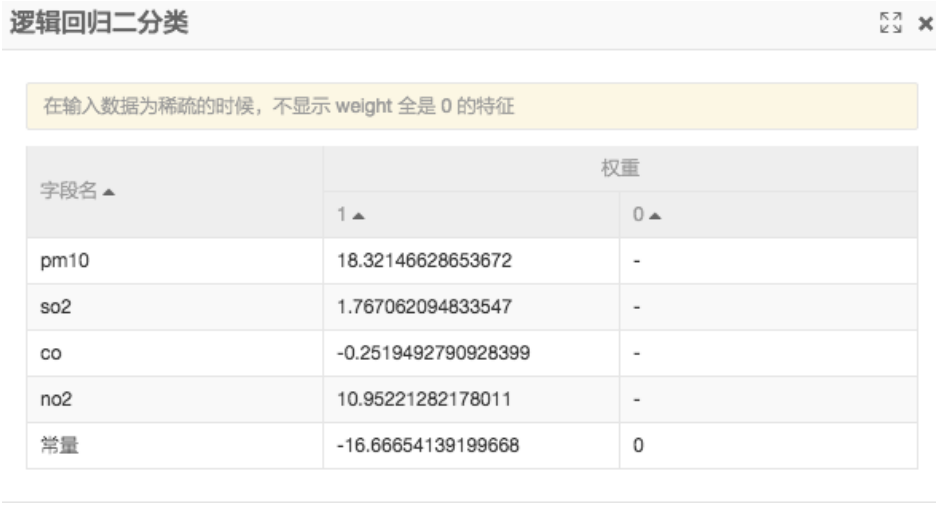


上图中的AUC为0.98，比用随机森林计算得到的结果略低一点。如果排除调参对于结果的影响，可以说明针对这个数据集，随机森林的训练效果会更好一些。

模型评估分析

根据上文中的模型和预测结果来分析哪种空气指标对于PM2.5影响最大。

逻辑回归生成的模型如下图所示。



逻辑回归二分类

在输入数据为稀疏的时候，不显示 weight 全是 0 的特征

字段名	权重	
	1	0
pm10	18.32146628653672	-
so2	1.767062094833547	-
co	-0.2519492790928399	-
no2	10.95221282178011	-
常量	-16.66654139199668	0

经过归一化计算的逻辑回归算法的模型系数越大，对于结果的影响越大。系数符号为正表示正相关，为负表示负相关。上图中正号系数里pm10和no2最大。

- pm10和pm2只是颗粒尺寸大小不同，是一个包含关系，可以不考虑。
- no2（二氧化氮）对于pm2.5的影响最大。查阅相关文档，了解哪些因素会造成no2的大量排放，即可找出影响pm2.5的主要因素。
  通过来自互联网的no2来源文章，说明了no2主要来自汽车尾气。

其它

请进入阿里云数加机器学习平台体验阿里云机器学习产品，并通过云栖社区公众号参与讨论。

Caffe实现图片分类

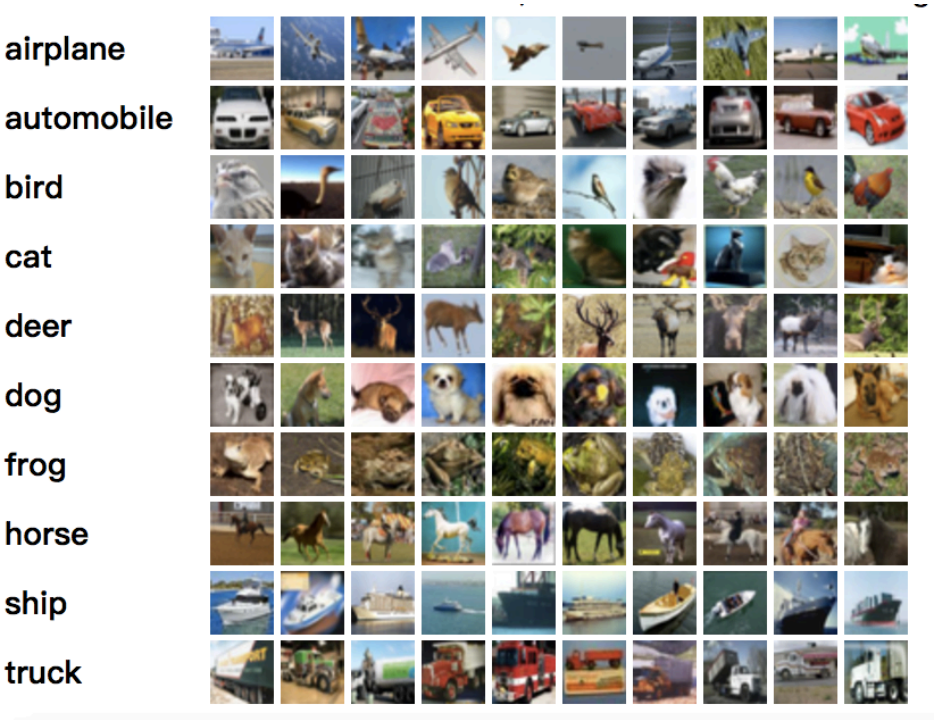
背景

TensorFlow实现图像分类文档介绍了如何通过深度学习的TensorFlow框架，实现对Cifar10图像的分类。
本文档介绍另一个深度学习框架Caffe，通过Caffe只需要填写一些配置文件就可以实现图像分类的模型训练。

请提前阅读深度学习文档，在机器学习平台上开通深度学习功能，文末提供了相关下载链接。

数据介绍

本文使用的是cifar10开源数据集，包含6万张像素为32*32的彩色图片，这6万张图片被分成10个类别，分别是飞机、汽车、鸟、毛、鹿、狗、青蛙、马、船、卡车，数据集截图如下。



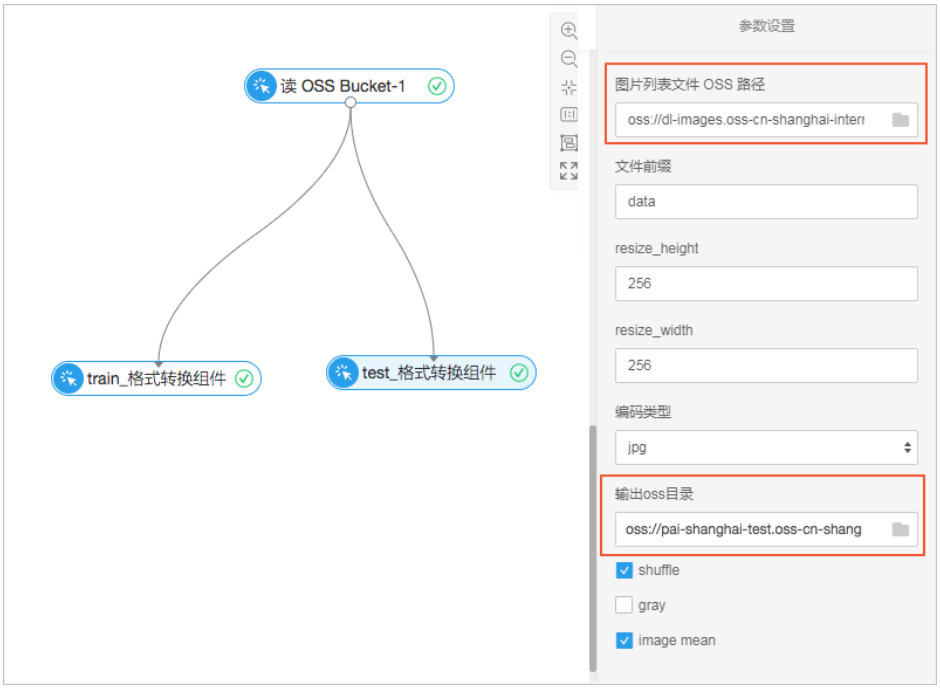
这份数据已经内置在机器学习平台的公共数据集中，以jpg格式存储。任何机器学习用户都可以在深度学习组件的**数据源目录**中直接输入以下路径：

- 测试数据： oss://dl-images.oss-cn-shanghai-internal.aliyuncs.com/cifar10/caffe/images/cifar10_test_image_list.txt
- 训练数据：oss://dl-images.oss-cn-shanghai-internal.aliyuncs.com/cifar10/caffe/images/cifar10_train_image_list.txt

如下图所示。

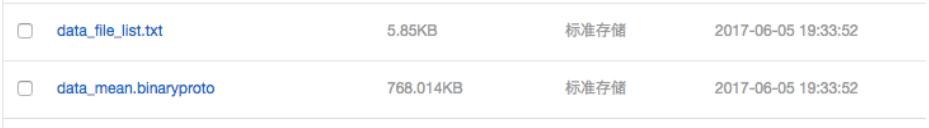
格式转换

目前深度学习的Caffe框架只支持特定的格式，所以首先需要使用“格式转换”组件，对jpg格式的图片进行转换。



- 图片列表文件OSS路径：上文提到的机器学习内置的公共数据集。
- 输出oss目录：用户自定义的OSS目录。

经过格式转换，在输出的OSS目录下生成如下文件，训练数据和测试数据各一份。



需要记录对应的OSS路径用于Net文件的填写，假设路径名分别是：

训练数据data_file_list.txt：bucket/cifar/train/data_file_list.txt

训练数据data_mean.binaryproto:bucket/cifar/train/data_mean.binaryproto

测试数据data_file_list.txt：bucket/cifar/test/data_file_list.txt

测试数据data_mean.binaryproto:bucket/cifar/test/data_mean.binaryproto

Caffe配置文件

Net文件编写，对应上文格式转换生成的路径：

```
  }
  transform_param {
    mean_file: "bucket/cifar/train/data_mean.binaryproto"
    crop_size: 31
  }
  binary_data_param {
    source: "bucket/cifar/train/data_file_list.txt"
    batch_size: 100
  }
}
layer {
  name: "cifar"
  type: "BinaryData"
  top: "data"
  top: "label"
  include {
    phase: TEST
  }
  transform_param {
    mean_file: "bucket/cifar/test/data_mean.binaryproto"
    crop_size: 31
  }
  binary_data_param {
    source: "bucket/cifar/test/data_file_list.txt"
    batch_size: 100
  }
```

Solver文件编写：

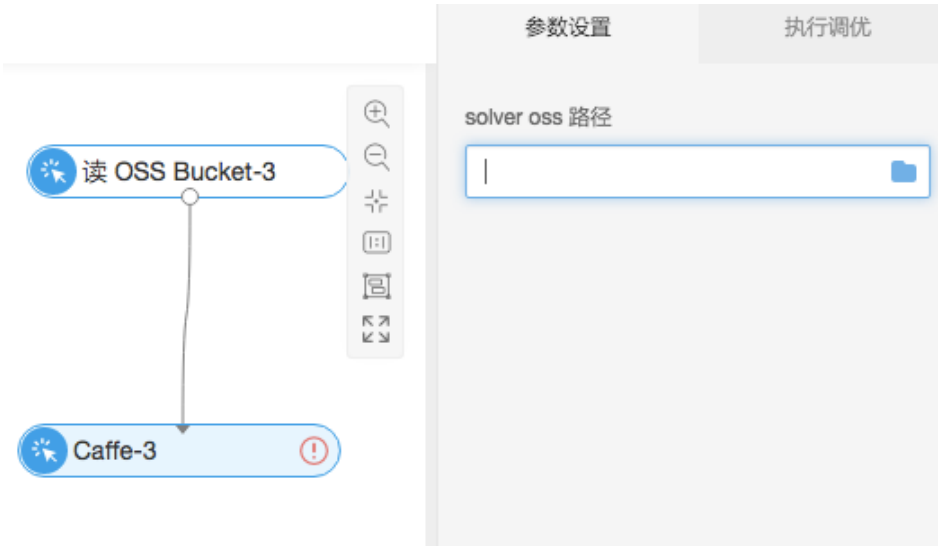
```
# The train/test net protocol buffer definition
net: "填写net文件的OSS路径"
# test_iter specifies how many forward passes the test should carry out.
# In the case of MNIST, we have test batch size 100 and 100 test iterations,
# covering the full 10,000 testing images.
test_iter: 100
# Carry out testing every 500 training iterations.
test_interval: 500
# The base learning rate, momentum and the weight decay of the network.
base_lr: 0.001
momentum: 0.9
weight_decay: 0.004
# The learning rate policy
lr_policy: "fixed"
# Display every 100 iterations
display: 100
# The maximum number of iterations
max_iter: 5000
# snapshot intermediate results
snapshot_after_train: true
# snapshot: 10000
# snapshot_format: HDF5
snapshot_prefix: "生成model的存储路径"
# solver mode: CPU or GPU
solver_mode: GPU
data_distribute_mode: MANUALLY
model_average_iter_interval: 1
```

运行

将编辑好的Solver文件和Net文件上传到OSS上，拖拽Caffe组件到画布中，并与数据源链接。

配置Caffe组件参数，如下图所示，**sovler oss路径**选择已经上传到OSS上的Solver文件，单击**运行**

。



在OSS的模型路径下查看生成的图片分类模型文件，结果如下，可以用以下模型进行图片分类。

参考TensorFlow实现图像分类的“日志查看”章节，查看日志。

开发者最佳实践

目录

利用PAI-DSW访问Github, 快速获取最新的学习资源

在学习数据科学的时候，我们往往需要从各个地方下载各种各样的数据集和代码。其中大名鼎鼎的就是我们的Github。这篇文章会简单讲讲我们如何从Github下载想要的学习资源，然后在DSW上进行运行和学习。

更详细的文章内容请查看用户在云栖社区的分享，查看更多>>

发电场输出电力预测

前言

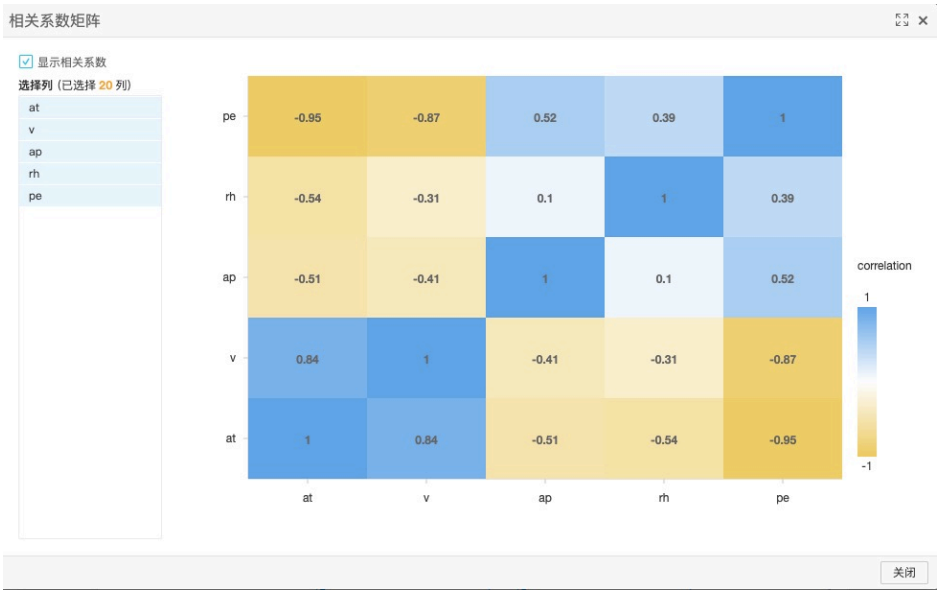
机器学习很多时候在工业场景下也会有非常好的应用。本次实验，我们就会以一个综合循环发电厂的发电数据来展示机器学习是如何应用到工业生产的实际场景中的。

本实验数据采集自 UCI 机器学习数据集中的 混合发电厂数据。对于发电厂来说，风力发电的输出电力很大情况下决定了单位发电机能够生产的电能。因此，通过收集系统各个相关指标来预测最终的输出电力对于发电厂来说是非常有帮助的。有效的预测发电机的输出电力可以更好的评估安排电力生产计划，避免资源的浪费。

载入数据并进行数据探索

载入好数据集之后，里面是一个综合循环发电场的数据，一共有9568个样本数据。每个数据有5列，分别为:AT（温度），V（压力），AP（湿度），RH（压强），PE（输出电力)。下面是数据预览的截图：

然后为了找出对 PE 输出电力影响最大的因素，我们可以从左侧 组件-统计分析 拖入**相关系数矩阵**这个组件，来观察各个特征对于输出电力。



右键单击完成的组件，选择查看分析报告，就可以得到我们的相关性分析了。从这张相关性图中，我们不难看到和 输出电力最相关的因素就是 温度，其次是 压力，然后是湿度，再然后是压强。

对数据进行建模

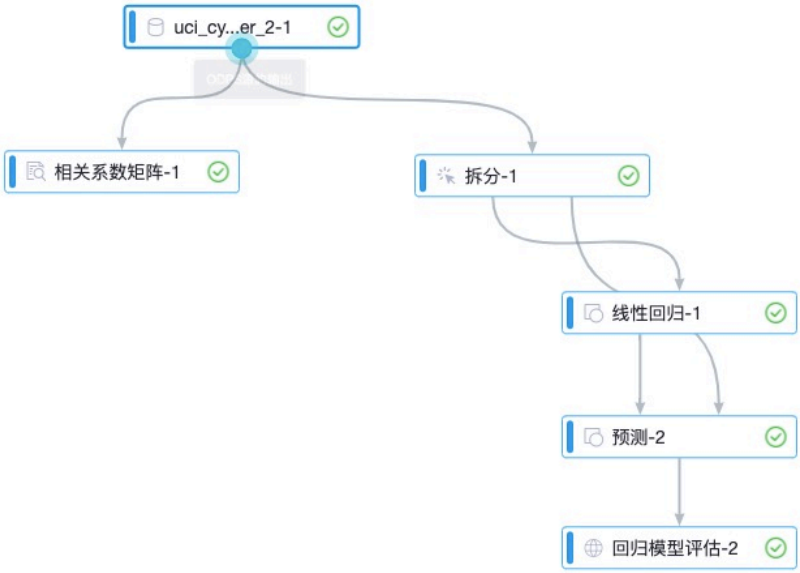
观察完数据相关性之后，我们可以通过 组件-数据预处理 中的**拆分**组件 对数据做一次拆分，将数据分为训练集和测试集。然后再使用 组件-机器学习-回归 中的**线性回归** 来对我们的数据进行回归建模。这里我们需要选择我们的特征列(X)和我们的标签列(Y)

对回归模型进行预测和评估

建模完毕之后，我们可以通过 组件-机器学习 中的**预测**来预测该模型在测试数据集上的效果。只需要进行如下的配置即可 特征列我们选择 at,v,ap,rh 原样输出列我们全选即可。

我们在这一步完成之后可以右键模型，点击查看模型 即可看到不同的特征对于我们的结果量的权重

最后，再从左侧的 组件-机器学习-评估 中选择**回归模型评估**即可获得我们的模型效果。右键 回归模型评估-查看分析报告 即可发现我们的 RMSE 到达了 4.57。下面是整个实验完成后的截图



这样我们就通过线性回归模型建立了一个混合发电厂的发电电力预测模型。通过模型部署之后，我们就可以实时的为发电厂提供发电电力的预估，以便更好的安排电力的生产计划，避免资源浪费。

用户窃电识别

传统的防窃漏电方法主要通过定期巡检、定期校验电表、用户举报窃电等方法来发现窃电或计量装置故障。但

这种方法对人的依赖性太强，抓窃查漏的目标不明确。

目前，很多供电局主要通过工作人员利用计量异常报警功能和电能量数据查询功能开展用户用电情况的在线监控工作，通过采集电量异常、负荷异常、终端报警、主站报警、线损异常等信息监测窃漏电情况和发现计量装置的故障。根据报警事件发生前后客户计量点有关的电流、电压、负荷数据情况等，构建基于指标加权的用电异常分析模型，实现检查客户是否存在窃电、违章用电．及计量装置故障等。

以上防窃漏电的诊断方法，虽然能获得用电异常的某些信息，但由于终端误报或漏报过多，无法达到真正快速精确定位窃漏电嫌疑用户的目的，往往令稽查工作人员无所适从。而且在采用这种方法建模时，模型各输人指标权重的确定需要用专家的知识和经验来判断，具有很大的主观性，存在明显的缺陷，所以实施效果往往不尽如人意。

现有的电力计量自动化系统能够采集到各相电流、电压、功率因数等用电负荷数据以及用电异常等终端报警信息。异常告警信息和用电负荷数据能够反映用户的用电情况，同时稽查工作人员也会通过在线稽查系统和现场稽查来找出窃漏电用户，并录入系统。

通过这些数据信息提取出窃漏电用户的关键特征，构建窃漏电用户的识别模型，就能自动检查、判断用户是否存在窃漏电行为,大大降低稽查工作人员的工作量，保障人民的正常用电，安全用电。

载入数据并进行数据探索

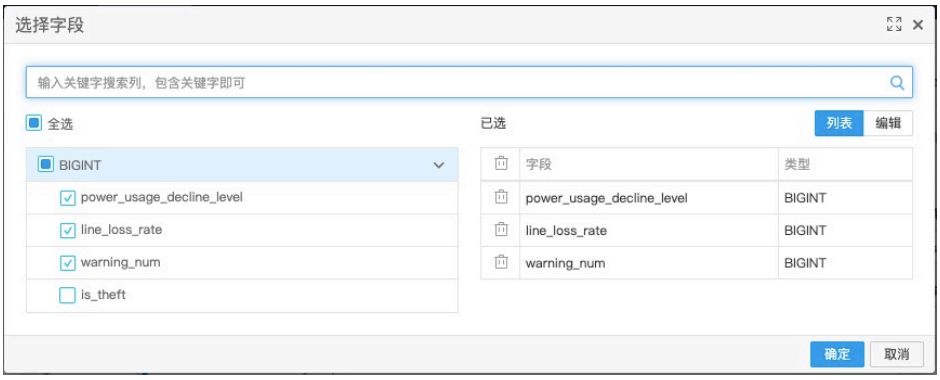
选择好数据集之后,里面是一个用户的三个窃漏电指标以及用户是否真实窃漏电的数据。其中包括：电量趋势下降指标,线损指标,告警类指标数量 以及 是否窃漏电。


数据探查 – anti_electricity_theft – (仅显示前一百条)

序号	power_usage_decline_level	line_loss_rate	warning_num	is_theft
1	4	1	1	1
2	4	0	4	1
3	2	1	1	1
4	9	0	0	0
5	3	1	0	0
6	2	0	0	0
7	5	0	2	1
8	3	1	3	1

这里我们可以通过从左侧 组件-统计分析 拖入**相关系数矩阵**这个组件，来观察各个特征对于输出电力。

右键单击完成的组件，选择查看分析报告，就可以得到我们的相关性分析了。从这张相关性图中，我们会发现，其实这三个指标对于最终是否为窃电用户的关系都不是特别明显，也就是说决定用户是否为窃电用户的特征并不明显的具有单一性。此时我们还可以通过左侧的 组件-统计分析 拖入**数据视图** 来分析各个特征对于我们的标签列的数据分布。我们只需要按照如下配置选择特征列



然后选择我们的标签列

最后我们同样右键单击从此处开始运行后，右键单击完成的组件，选择查看分析报告，就可以看到各个特征和标签列在数据分布上的关系。

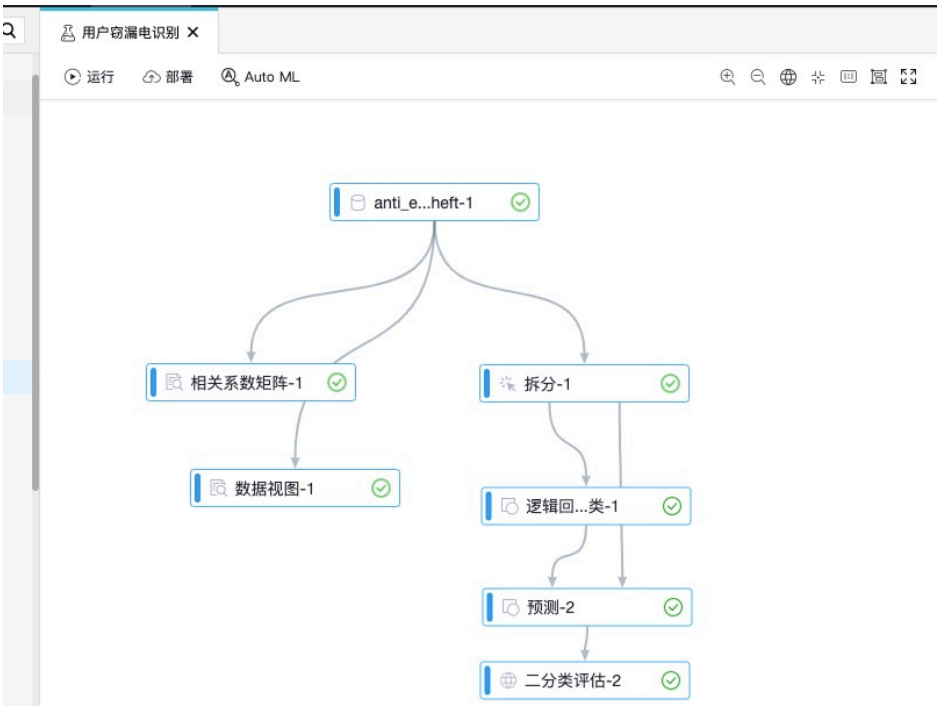
对数据进行建模

完成简单的探索性分析之后，我们就可以开始选择合适的算法模型建模了。我们可以先通过 组件-数据预处理 中的**拆分**组件 对数据做一次拆分，将数据分为训练集和测试集。

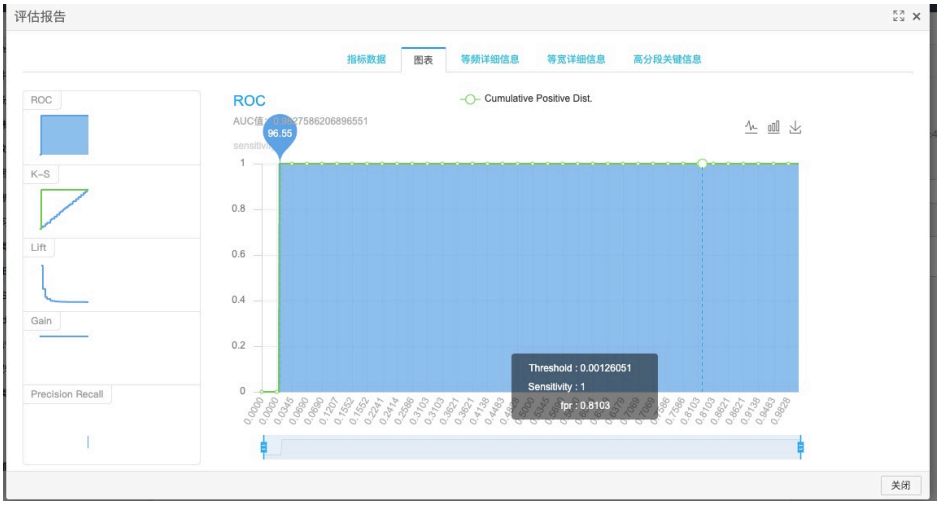
然后我们可以使用组件-机器学习-回归 中的**逻辑回归二分类** 来对我们的数据进行回归建模。这里我们需要选择我们的特征列(X)和我们的标签列(Y) 这里我们的特征列就选择：power_usage_decline_level,line_loss_rate 和 warning_num

对回归模型进行预测和评估

建模完毕之后，我们可以通过 组件-机器学习 中的**预测**来预测该模型在测试数据集上的效果。特征和原样输出我们都可以默认全选。然后我们再从左侧的 组件-机器学习-评估 中选择**二分类评估** 即可获得我们的模型效果这时候整个实验的应该如下图:



右键我们完成运行的二分类评估组件，即可看到我们的模型效果。这里我们的AUC达到了 0.9827, 效果非常不错。



这样我们就通过机器学习PAI平台完成了用户窃电行为的识别。我们可以通过EAS在线部署将这个服务部署为可在线调用的服务，为电网提供用户窃电行为的在线识别服务.

本实验参考了《Python数据分析与挖掘实战》，如有版权等问题，请联系本文作者。我们尊重学术领域每一位研究者们对于学术的贡献，致力将技术和现实生活更好的结合应用落地。