Machine Learning Platform for AI

Best Practices

Best Practices

Heart disease prediction

Overview

Heart disease is the biggest killer of humans. Heart disease causes 33% of deaths in the world. In China, hundreds and thousands of people die of heart disease every year. Data mining has become extremely important for heart disease prediction and treatment. It uses the relevant health exam indicators and analyzes their influences on heart disease. This document introduces how to use Alibaba Cloud Machine Learning Platform for AI to create a heart disease prediction model based on the data collected from heart disease patients.

Datasets

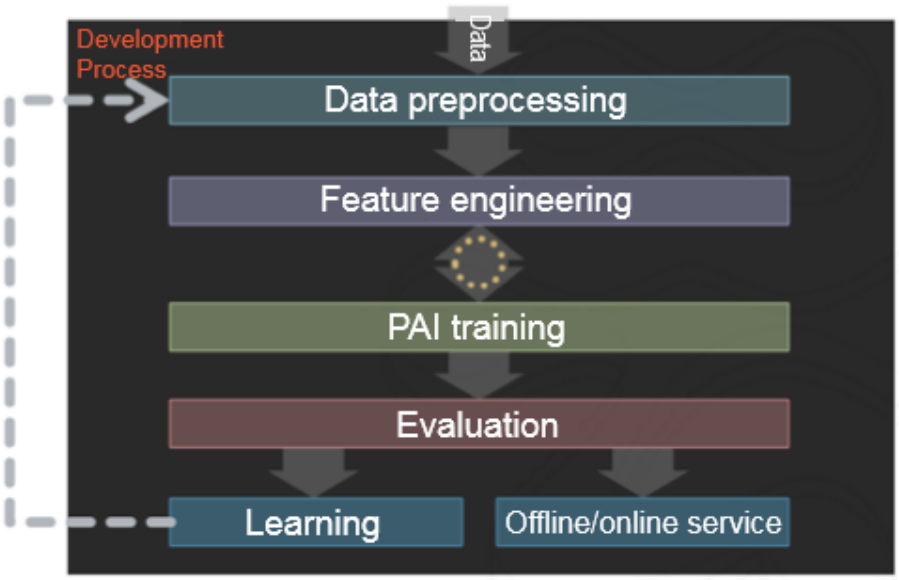
Data source UCI Heart Disease Dataset. This dataset is created based on 303 cases of heart disease in the United States. The attributes are as follows:

Name	Definition	Data Type	Description
age	Age	string	Age of a patient. The age attribute only uses numbers.
sex	Gender	string	Gender of a patient: female or male.
cp	Chest pain type	string	Chest pain types, including typical, atypical, non-anginal, and asymptomatic.
trestbps	Blood pressure	string	Blood pressure of a patient.
chol	cholesterol	string	Cholesterol of a patient.
fbs	Fasting blood sugar	string	True means that a patient’ s fasting blood sugar is

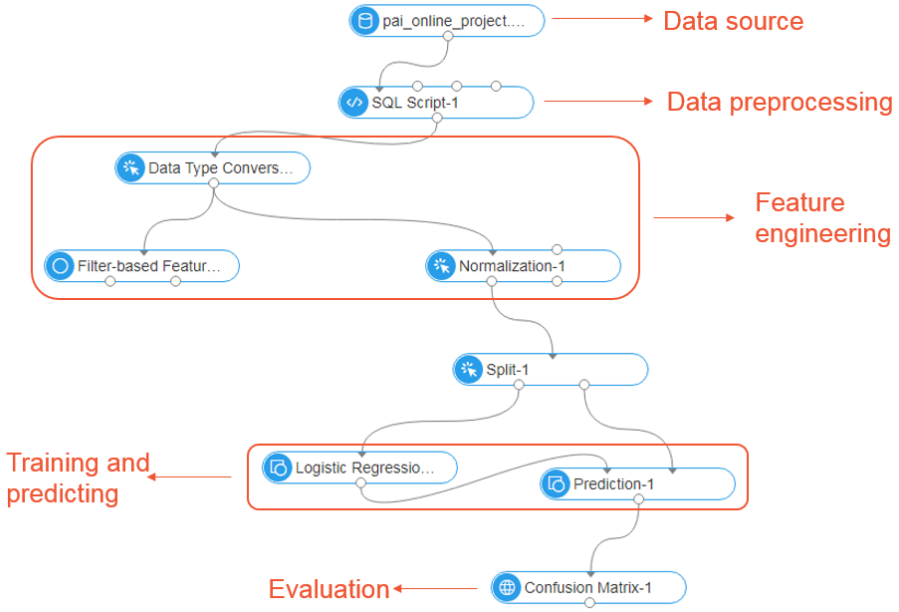
			greater than 120 mg/dl. False means that a patient’ s fasting blood sugar is equal to or less than 120 mg/dl.
restecg	Resting electrocardiographic result	string	The resting electrocardiographic results include normal, having ST-T wave abnormality, and showing probable or definite left ventricular hypertrophy.
thalach	Maximum heart rate achieved	string	Maximum heart rate of a patient.
exang	Exercise induced angina	string	True means that a patient has exercise induced angina. False means that a patient does not have exercise induced angina.
oldpeak	ST depression induced by exercise relative to rest	string	ST depression of a patient.
slop	Slope of the peak exercise ST segment	string	Slopes of the peak exercise ST segment, including down, flat, and up.
ca	Number of major vessels colored by flouroscopy	string	Number of major vessels colored by flouroscopy
thal	Defect type	string	defect types, including norm, fix, and rev.
status	Heart disease status	string	Health means that a patient does not have heart disease. Sick means that a patient has heart disease.

Data exploring procedure

The following figure shows the procedure of data mining:



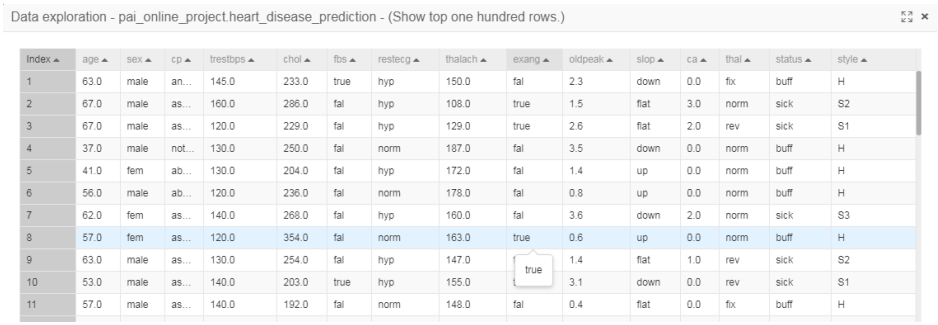
The following figure shows the workflow of the project:



Data pre-processing

Data pre-processing, also known as data cleaning, is the process of analyzing and making changes to the source data, including irrelevant data removal, incomplete data fixing, and data type conversion. With 14 indicators and one goal field, this project focuses on predicting the presence or absence of heart disease in patients based on their health exam indicators. The project uses one of the generalized linear models: logistic regression. Additionally, the data type of all input indicators is double.

All input data:


Data exploration - pai_online_project.heart_disease_prediction - (Show top one hundred rows.)

Index	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slop	ca	thal	status	style
1	63.0	male	an...	145.0	233.0	true	hyp	150.0	fal	2.3	down	0.0	fix	buff	H
2	67.0	male	as...	160.0	286.0	fal	hyp	108.0	true	1.5	flat	3.0	norm	sick	S2
3	67.0	male	as...	120.0	229.0	fal	hyp	129.0	true	2.6	flat	2.0	rev	sick	S1
4	37.0	male	not...	130.0	250.0	fal	norm	187.0	fal	3.5	down	0.0	norm	buff	H
5	41.0	fem	ab...	130.0	204.0	fal	hyp	172.0	fal	1.4	up	0.0	norm	buff	H
6	56.0	male	ab...	120.0	236.0	fal	norm	178.0	fal	0.8	up	0.0	norm	buff	H
7	62.0	fem	as...	140.0	268.0	fal	hyp	160.0	fal	3.6	down	2.0	norm	sick	S3
8	57.0	fem	as...	120.0	354.0	fal	norm	163.0	true	0.6	up	0.0	norm	buff	H
9	63.0	male	as...	130.0	254.0	fal	hyp	147.0	t	1.4	flat	1.0	rev	sick	S2
10	53.0	male	as...	140.0	203.0	true	hyp	155.0	t	3.1	down	0.0	rev	sick	S1
11	57.0	male	as...	140.0	192.0	fal	norm	148.0	fal	0.4	flat	0.0	fix	buff	H

true

During data pre-processing, we must convert data of string and text types to numeric type based on the definition of the data.

Boolean data
For example, you can set the sex attribute to 0 to indicate female and set the attribute to 1 to indicate male.

Multivalued data
For example, you can use 0 through 3 to numerically rate the chest pain in ascending order for the cp attribute.

The data pre-processing is based on SQL scripts. Learn more, see the SQL script-1 component as follows:

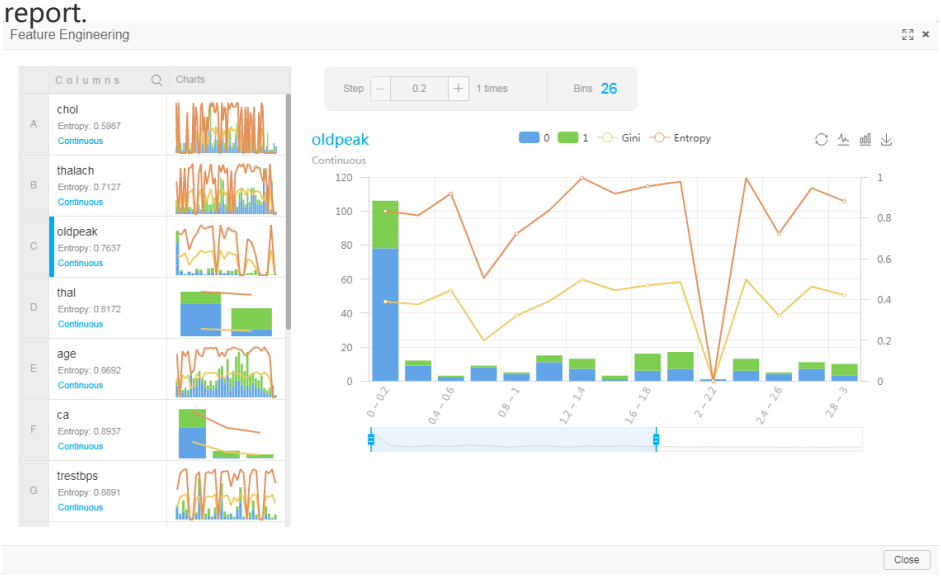
```
select age,
(case sex when 'male' then 1 else 0 end) as sex,
(case cp when 'angina' then 0 when 'notang' then 1 else 2 end) as cp,
trestbps,
chol,
(case fbs when 'true' then 1 else 0 end) as fbs,
(case restecg when 'norm' then 0 when 'abn' then 1 else 2 end) as restecg,
thalach,
(case exang when '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};
```

Feature engineering

Feature engineering includes feature derivation and scale change. This project uses the feature selection and data normalization components for feature engineering.

Filter-based feature selection

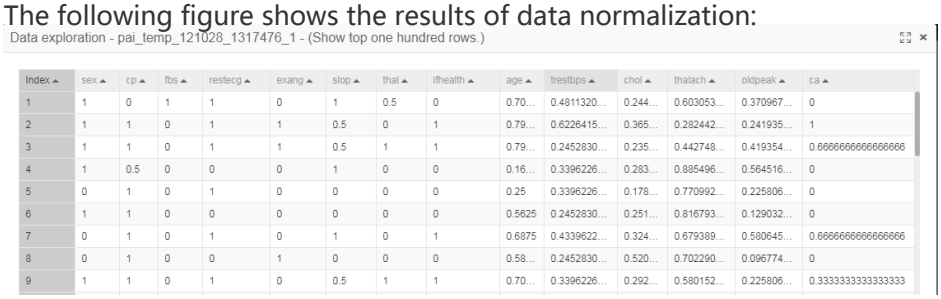
This component measures the influence of each indicator on the prediction results by using the entropy and Gini coefficient. You can view the final prediction results in the assessment report.



Data normalization

This project requires you to train your model by using dichotomous logistic regression. Therefore, you must avoid using different fundamental units for the indicators. Data normalization uses the following formula to ensure that all indicators use a value between 0 and 1: result = (val-min) / (max-min).

The following figure shows the results of data normalization:



Index	sex	cp	fbs	restecg	exang	slop	thal	ifhealth	age	trestbps	chol	thalach	oldpeak	ca
1	1	0	1	1	0	1	0.5	0	0.70...	0.4811320...	0.244...	0.603053...	0.370967...	0
2	1	1	0	1	1	0.5	0	1	0.79...	0.6226415...	0.365...	0.282442...	0.241935...	1
3	1	1	0	1	1	0.5	1	1	0.79...	0.2452830...	0.235...	0.442748...	0.419354...	0.6666666666666666
4	1	0.5	0	0	0	1	0	0	0.16...	0.3396226...	0.283...	0.885496...	0.564516...	0
5	0	1	0	1	0	0	0	0	0.25	0.3396226...	0.178...	0.770992...	0.225806...	0
6	1	1	0	0	0	0	0	0	0.5625	0.2452830...	0.251...	0.816793...	0.129032...	0
7	0	1	0	1	0	1	0	1	0.6875	0.4339622...	0.324...	0.679389...	0.580645...	0.6666666666666666
8	0	1	0	0	1	0	0	0	0.58...	0.2452830...	0.520...	0.702290...	0.096774...	0
9	1	1	0	1	0	0.5	1	1	0.70...	0.3396226...	0.292...	0.580152...	0.225806...	0.3333333333333333

Model training and prediction

Supervised learning requires you to train your model to obtain the prediction results and compare the prediction results with the existing data. In this project, supervised learning is used to train the model to predict the presence or absence of heart disease in a group of patients.

Data split

Use the split component to split the data into the training dataset and predicting dataset at the ratio of 7:3. The training dataset is imported to the dichotomous logistic regression component for model training. The predicting dataset is imported to the prediction

component.

Dichotomous logistic regression

Logistic regression is a linear model. In this project, dichotomous logistic regression (determining the presence or absence of heart disease) is achieved by comparing the prediction results with a threshold. You can learn more about logistic regression from the Internet or relevant documentation. You can view the model that has already been trained by logistic regression on the Model page.

Logistic Regression Output

feature	weight	
	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	-

Prediction

The prediction component has two inputs: the model and the predicting dataset. The prediction results show the calculated data, the predicting data, and the probability of inconsistencies between the calculated data and predicting data.

Assessment

You can use the confusion matrix to assess the attributes of the model, such as the accuracy.

Confusion Matrix

Confusion Matrix | Proportion Matrix | Stats

Models	true count	False count	Summary	Accuracy	Precision	Recall Rate	F1
0	47	11	58	82.418%	81.034%	90.385%	85.455%
1	28	5	33	82.418%	84.848%	71.795%	77.778%

Based on the accuracy of the prediction result, you can determine whether your model is well trained or not.

Conclusions

According to the workflow of data exploring, the following conclusions can be made:

Feature weight

- You can obtain the weight of each indicator in the prediction by using filter-based feature selection.

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

- The maximum heart rate achieved (thalach) indicator has the greatest impact on heart disease prediction.
- The gender indicator does not have any impact on heart disease prediction.

Prediction results

Based on the 14 indicators, the model can predict heart disease with an accuracy of over 80%. This model can be used in heart disease prediction and treatment.

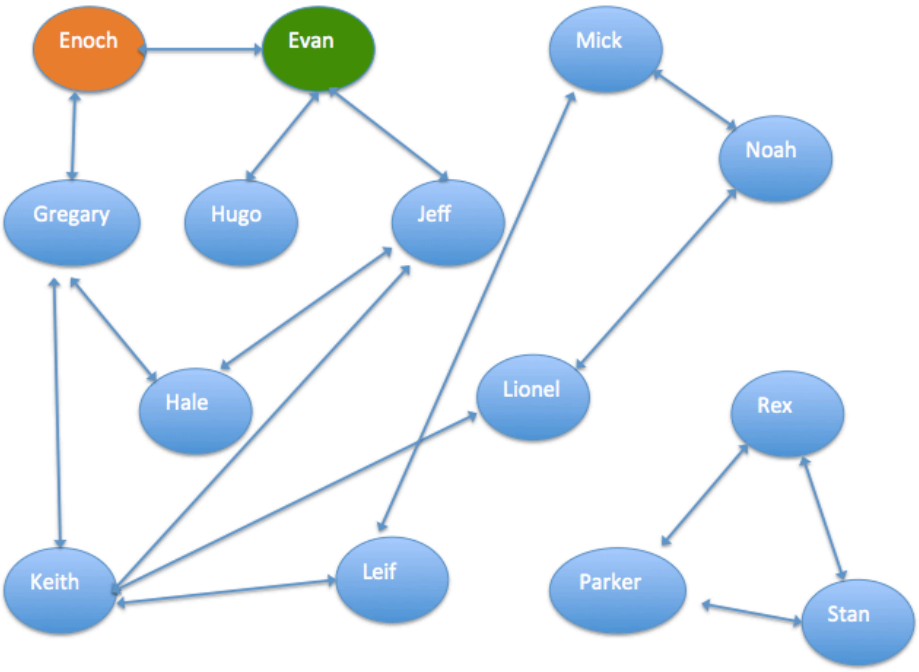
Financial risk management

Overview

This project is created by using Alibaba Cloud Network Chart. Network Chart is used to illustrate the interconnections among a set of entities, for example, the relationships among a group of people. Unlike hierarchical data, the relationships in Network Chart are represented by nodes and edges (links). The nodes are connected to each other through edges. Alibaba Cloud Machine Learning Platform For AI provides several Network Chart components, including K-Core, largest connected subgraph, and label propagation classification.

Scenario

The following figure shows the relationships among a group of people. The arrows in the figure represent the relationships between these people (for example, coworkers or relatives). Enoch is a trusted customer and Evan is a fraudulent customer. Based on this information and the relationship graph, Network Chart allows you to calculate the credit scores of the remaining people for financial risk management. By referencing the credit scores, you can make predictions about which of them may be fraudulent customers.



Datasets

Data source: the dataset in this project is provided by Alibaba Cloud Machine Learning Platform For AI. The dataset includes the following attributes:

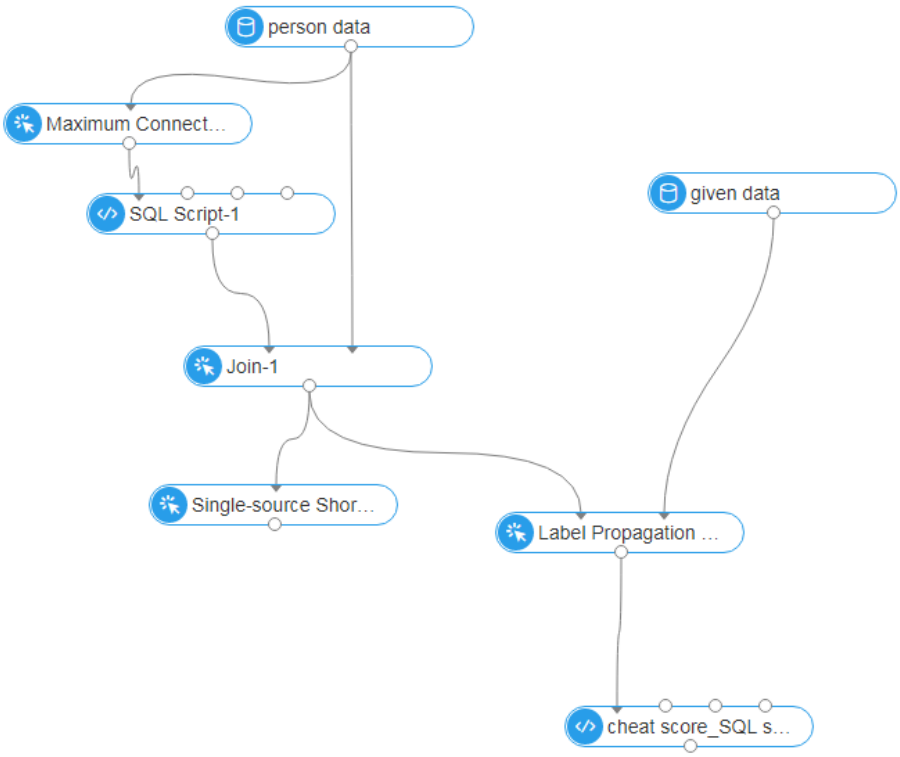
Name	Definition	Data Type	Description
start_point	Start node of an edge	string	Name of a person.
end_point	End node of an edge	string	Name of a person.
count	Relational closeness	double	The larger the value is, the closer relationship the two persons have.

The following figure shows the data entries:

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

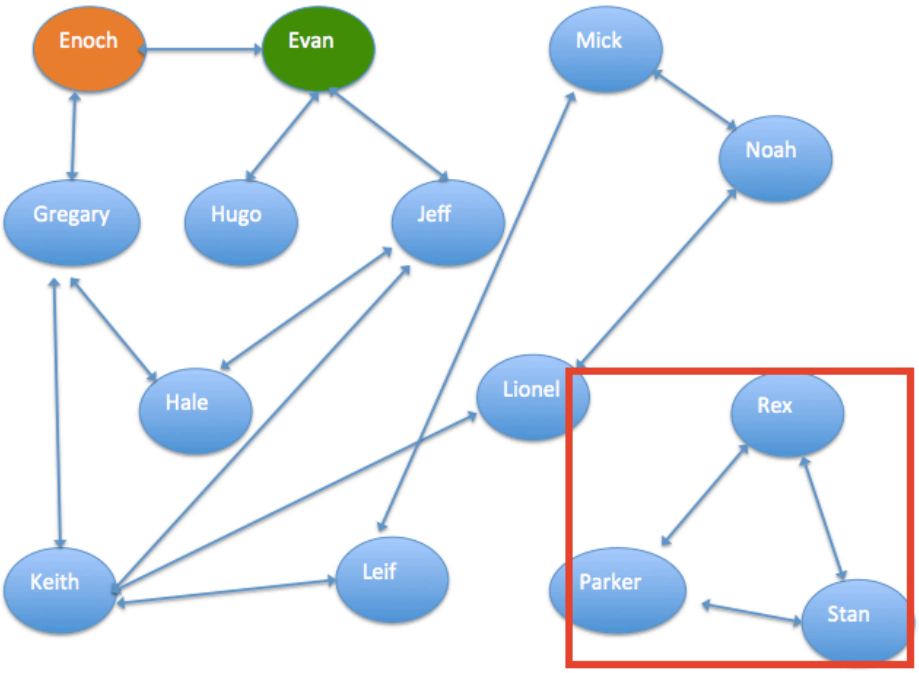
Data exploring procedure

The following figure shows the workflow of this project:



Largest connected subgraph

The largest connected subgraph allows you to find the cluster that contains the most interconnected entities. In this project, the largest connected subgraph divides the people into two groups and assigns each team a group ID (group_id). The group containing Parker, Rex, and Stan should be removed from the subgraph because the relationship between these people do not affect the prediction results. You can use the SQL script component and JOIN component to remove this group from the subgraph.



Single-source shortest path

The single-source shortest path allows you to measure the distance (number of nodes) that a start node must pass through to reach an end node.

The following figure shows the distances between Enoch and the others:

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

Label propagation classification

Label propagation classification is a semi-supervised classification algorithm. It uses the existing label information of the nodes to predict the label information of the unlabeled nodes. Based on the correlations between the nodes, label propagation classification propagates each label to other nodes.

To use the label propagation classification component, make sure that you have a connected graph containing all entities and the data for labeling. In this project, the data for labeling is imported from the **Read Data Source** component. The weight column shows the probability of a person being a fraudulent customer.

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

By SQL filtering, the final results show the probabilities of committing fraud for all people. The larger the value is, the larger probability a person may be fraudulent customer.

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

Product recommendation

Overview

The parable of beer and diapers is a classic case of data mining utilization. The diapers and beer are irrelevant. However, when the diapers and beer are put next to each other on shelves, both of their sales increase. The problem is how to find the hidden correlation between two irrelevant products. To resolve this problem, you can use collaborative filtering, which is one of the algorithms commonly used in data mining. This algorithm enables you to find the hidden correlation between different customers and products.

Collaborative filtering is a correlation rule-based algorithm. This project takes shopping behaviors as an example, including customers A and B and products X, Y, and Z. If both customers A and B have purchased products X and Y, collaborative filtering determines that customers A and B have similar interests in shopping. Collaborative filtering then recommends product Z to customer B because customer A has purchased product Z. In this case, collaborative filtering works based on customers' interests.

Scenario:

This project shows how to use the customer shopping behaviors recorded before July to find the correlations between products. We then use this information to recommend relevant products to customers. In addition, the project also makes an assessment of the recommendation results. For example, customer A purchased product X before July. Product X is strongly correlated with product Y. The system then recommends product Y to customer A after July and calculates the probability of customer A purchasing product Y.

Datasets

Data source: the two datasets are provided by the Tianchi challenges, including the shopping behaviors before July and the shopping behaviors after July.

The attributes are as follows:

Name	Definition	Data Type	Description
user_id	User ID	string	User ID of a customer.
item_id	Product ID	string	ID of a product.
active_type	Shopping behavior	string	A value of 0 indicates that the product page is viewed by the customer. A value of 1 indicates that the product is purchased. A value of 2 indicates that

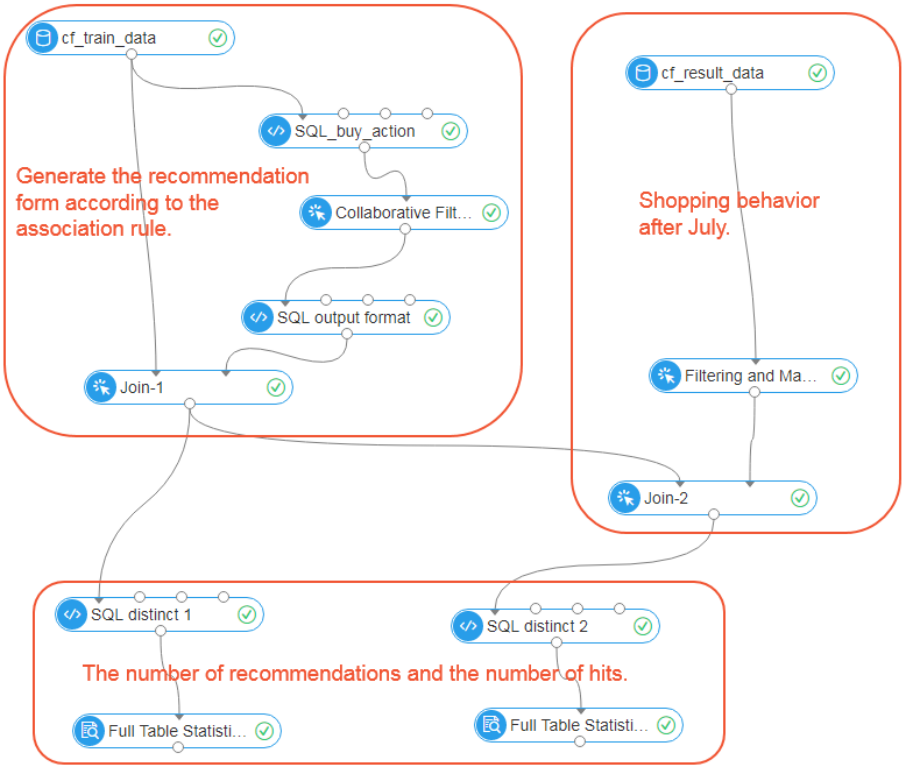
			the product is added to the customer’ s favorites. A value of 3 indicates that the product is added to the customer’ s shopping cart.
active_date	Purchased at	string	Time when the product is purchased.

The following figure shows the data entries:

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

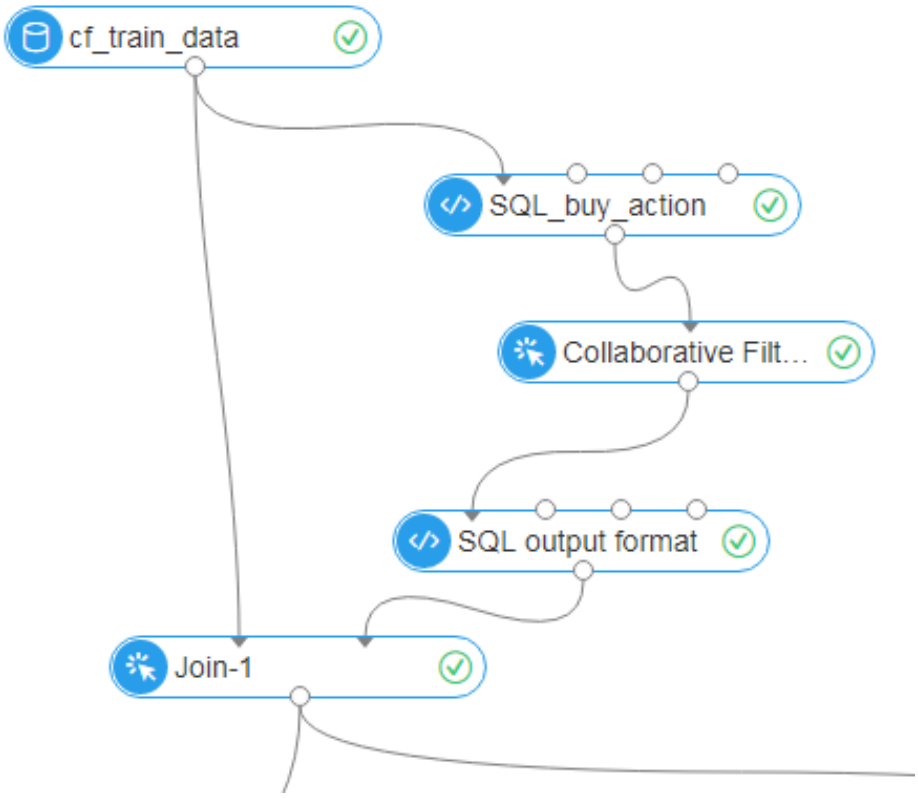
Data exploring procedure

The following figure shows the workflow of this project:

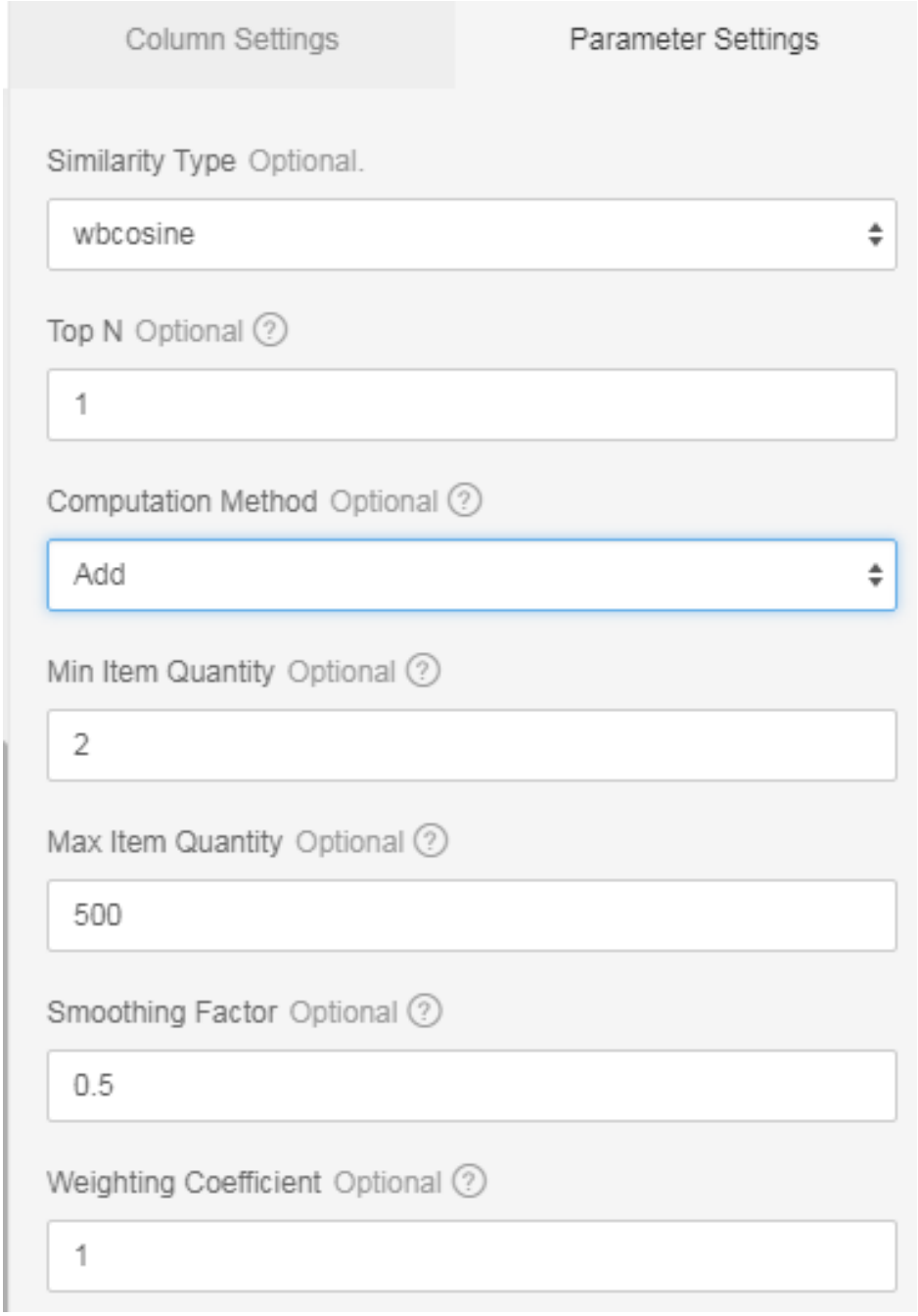


Collaborative filtering-based recommendation procedure

Load the dataset recorded before July, use SQL scripts to extract the shopping behaviors, and import the data to the collaborative filtering component. Set the **TopN** attribute to 1 for the collaborative filtering component. This allows the collaborative filtering component to find the most similar item for each input item and calculate its weight. Analyze the shopping behaviors and then make predictions about items that are most likely to be purchased by the same customer.



The following figure shows the relevant settings:



The following figure shows the collaborative filtering results. The **itemid** column shows the IDs of the target products. The **similarity** column shows two colon-separated items: ID of the product that is strongly correlated with the target product and the probability of this product being purchased.

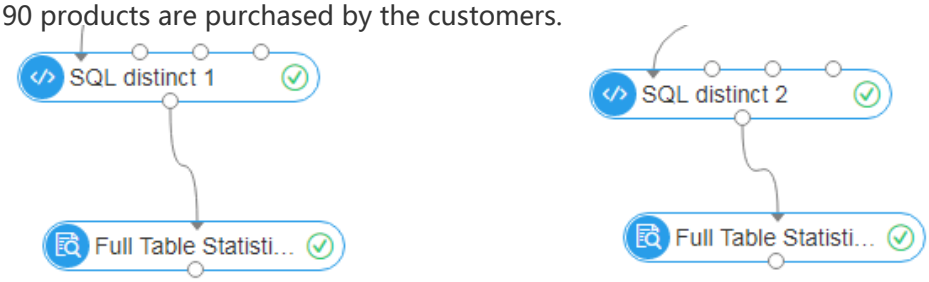
itemid ▲	similarity ▲
1000	15584:0.2747133918
10014	18712:0.05229603127
10066	3228:0.2650900672
1008	24507:1
10082	18024:0.1781525919
1010	18024:0.2104947227
10133	14020:0.2070609237
1015	18024:0.2104947227
10151	26288:0.4366713611
10171	11080:0.2401992435

Product recommendations

The preceding steps show how to list all strongly correlated products. The following figure shows the workflow of using the product similarity list to make recommendations and predicting the recommendation results. For example, if customer A purchased product X and product X is strongly correlated with product Y, product Y then is recommended to customer A.

Recommendation results

This figure shows the statistics components. The full table scan component 1 shows the recommendation list created based on the shopping behaviors before July. By removing any duplicate rows, the final list contains 18,065 entries. The full table scan component 2 shows the number of products (in the recommendation list) that are purchased by the customers. In this project, 90 products are purchased by the customers.



Conclusions

By referencing the recommendation results, we can still make the following improvements to the project:

The project should include all factors that may influence the recommendation results. For example, the shopping behaviors must be time effective. In this project, the dataset includes shopping behaviors recorded in several months. Using outdated data may prevent you from getting the expected recommendation results. Additionally, the project only focuses on the hidden correlations between the products. The attributes of the recommended products are not taken into consideration. For example, whether the products are frequently rated products or not. If customer A bought a cell phone last month, he may not buy another cell phone the next month. In this case, cell phones are infrequently rated products.

To increase the accuracy of the prediction, this project should use a model trained by machine learning. The latent product associations should be only used as supplementary data.

Credit card bill statements-based-credit scorecard

Overview

Scorecard is not only a machine learning algorithm, but also a generic modeling framework used to

build a model for assessing credit risks. In scorecard modeling, the original data is processed by data binning and feature engineering, and then is used to build a linear model.

Scorecard modeling is typically used in credit assessment scenarios, such as for credit card applications and loan disbursements. It is also used in other industries for scoring, including customer service scoring and Alipay credit scoring. This project shows how to use the financial component on Alibaba Cloud Machine Learning Platform for AI to build a scorecard model.

Datasets

The following dataset contains client information, such as gender, education, marital status, and age, payment history, and credit card billing statements. The payment_next_month column (goal field) indicates the probability of a client paying off their credit card debt, as shown in the following figure. A value of 1 indicates that the client will likely pay off the debt and a value of 0 indicates that the client will not likely pay off the debt.

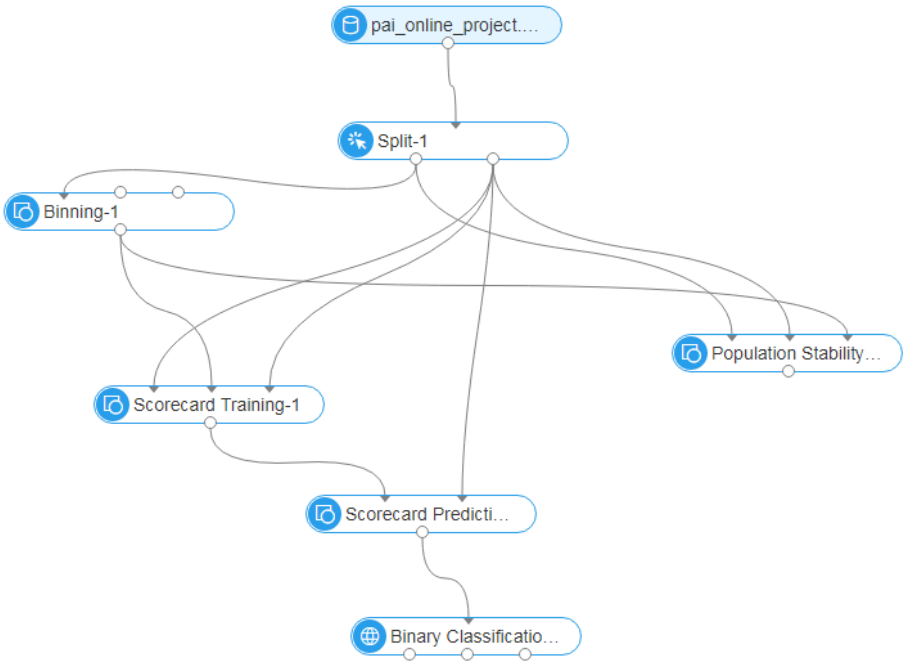
Source Table Columns

Columns	Type	Range from or
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

The dataset contains 30,000 entries. You can download the dataset from https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset.

Project workflow

The following figure shows the workflow of this project:



The procedure includes the following major steps:

Data split

Split the input data into two parts: one for model training and one for prediction result assessment.

Data binning

Data binning is similar to onehot encoding. It is a process of grouping the input data into data classes (bins). The data values in each bin are replaced by a value, which is the representative of the bin. As shown in the following figure, the binning component groups the age values into a number of age intervals:

	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					-	-	-	-	-	-

As shown in the following figure, after data binning, each field falls into multiple intervals:

Data exploration - pai_temp_121044_1317575_1 - (Show top one hundred rows.)

Index	feature	json
1	limit_bal	{"bin": {"norm": [{"iv": 0.07123500000000001, "n": 2086, "p": 1155, "prate": 0.356371, "total": 3241, "value": "(-inf,30000]", "woe": 0.669669}, {"iv": 0.011173, "n": 2074, "p":...
2	age	{"bin": {"norm": [{"iv": 0.008099, "n": 2257, "p": 816, "prate": 0.265539, "total": 3073, "value": "(-inf,25]", "woe": 0.243439}, {"iv": 0.0004929999999999999, "n": 1744, "p": 4...
3	pay_0	{"bin": {"norm": [{"iv": 0.047537, "n": 5746, "p": 1052, "prate": 0.154751, "total": 6798, "value": "(-inf,-1]", "woe": -0.436994}, {"iv": 0.170212, "n": 10241, "p": 1515, "prate": ...
4	pay_2	{"bin": {"norm": [{"iv": 0.007126, "n": 2490, "p": 552, "prate": 0.18146, "total": 3042, "value": "(-inf,-2]", "woe": -0.245673}, {"iv": 0.031622, "n": 4077, "p": 758, "prate": 0.156...
5	pay_3	{"bin": {"norm": [{"iv": 0.007195, "n": 2680, "p": 599, "prate": 0.182678, "total": 3279, "value": "(-inf,-2]", "woe": -0.237494}, {"iv": 0.034982, "n": 4025, "p": 728, "prate": 0.15...
6	pay_4	{"bin": {"norm": [{"iv": 0.005376, "n": 2853, "p": 664, "prate": 0.188797, "total": 3517, "value": "(-inf,-2]", "woe": -0.197027}, {"iv": 0.029571, "n": 3822, "p": 711, "prate": 0.15...
7	pay_5	{"bin": {"norm": [{"iv": 0.004848, "n": 2950, "p": 696, "prate": 0.190894, "total": 3646, "value": "(-inf,-2]", "woe": -0.183394}, {"iv": 0.027291, "n": 3748, "p": 707, "prate": 0.15...
8	pay_6	{"bin": {"norm": [{"iv": 0.003858, "n": 3166, "p": 767, "prate": 0.195017, "total": 3933, "value": "(-inf,-2]", "woe": -0.156921}, {"iv": 0.020181, "n": 3839, "p": 774, "prate": 0.16...
9	bill_amt1	{"bin": {"norm": [{"iv": 0.001837, "n": 1813, "p": 587, "prate": 0.244583, "total": 2400, "value": "(-inf,267]", "woe": 0.133103}, {"iv": 1e-06, "n": 1871, "p": 529, "prate": 0.2204...
10	bill_amt2	{"bin": {"norm": [{"iv": 0.000424, "n": 1945, "p": 587, "prate": 0.231833, "total": 2532, "value": "(-inf,0]", "woe": 0.062824}, {"iv": 5.1e-05, "n": 1777, "p": 492, "prate": 0.2168...

Population stability index

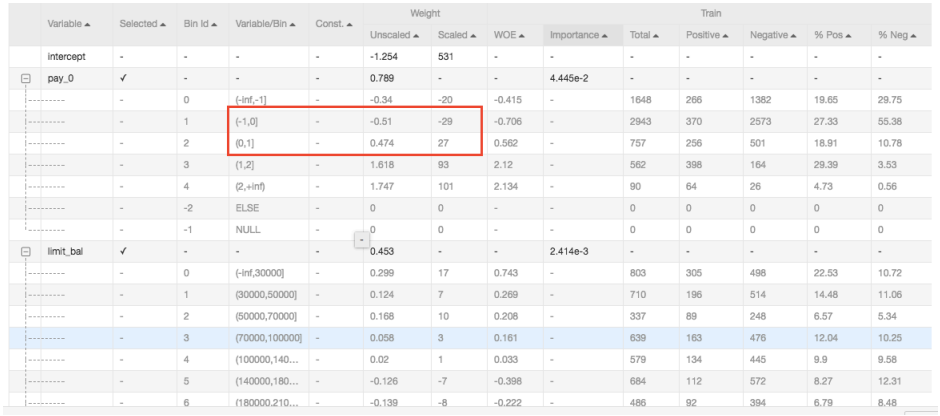
Population stability index (PSI) is an important metric to identify a shift in the population for credit scorecards, for example, the changes in the population within two months. A PSI value smaller than 0.1 indicates insignificant changes. A PSI value between 0.1 and 0.25 indicates minor changes. A PSI value larger than 0.25 indicates major changes in the population.

By comparing the stability of the population before data split, after data split, and after data binning, the model calculates the final PSI values for all features as follows:

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

Scorecard training

The following figure shows the scorecard training results:



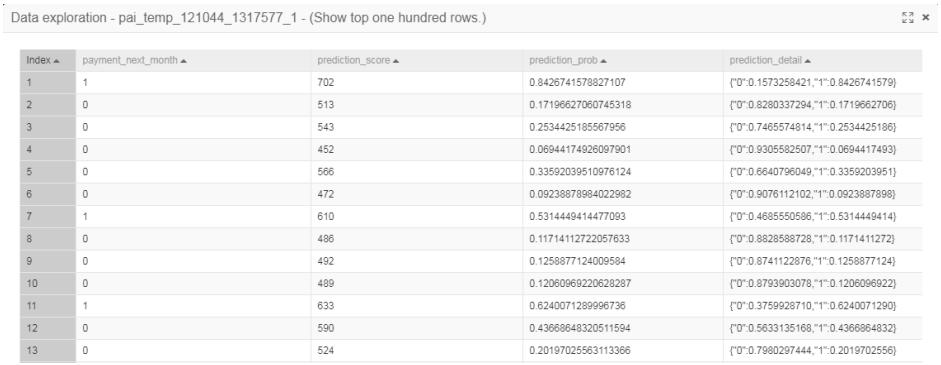
	Variable	Selected	Bin Id	Variable/Bin	Const.	Weight		WOE	Importance	Train				
						Unscaled	Scaled			Total	Positive	Negative	% Pos	% Neg
	intercept	-	-	-	-	-1.254	531	-	-	-	-	-	-	-
⊟	pay_0	✓	-	-	-	0.789	-	-	4.445e-2	-	-	-	-	-
	---------	-	0	(-inf,-1]	-	-0.34	-20	-0.415	-	1648	266	1382	19.65	29.75
	---------	-	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

The purpose of using the scorecard is to use normalized scores to indicate the weights of the features in the model.

- Unscaled: represents the original weight.
-
- Scaled: an index that indicates the amount of points that a feature gains or loses. For example, if the pay_0 feature falls into the (-1,0] bin, the feature gains 29 points. If the pay_0 feature falls into the (0,1] bin, the feature loses 27 points.
-
- Importance: represents the influence of each indicator on the prediction results. The larger the value is, the greater influence the indicator has.

Modeling results

In this project, the modeling results refer to the credit scores calculated for all clients, as shown in the following figure:



Data exploration - pai_temp_121044_1317577_1 - (Show top one hundred rows.)

Index	payment_next_month	prediction_score	prediction_prob	prediction_detail
1	1	702	0.8426741578827107	{"0":0.1573258421,"1":0.8426741579}
2	0	513	0.17196627060745318	{"0":0.8280337294,"1":0.1719662706}
3	0	543	0.2534425185567956	{"0":0.7465574814,"1":0.2534425186}
4	0	452	0.06944174926097901	{"0":0.9305582507,"1":0.0694417493}
5	0	566	0.33592039510976124	{"0":0.6640796049,"1":0.3359203951}
6	0	472	0.09238878984022982	{"0":0.9076112102,"1":0.0923887898}
7	1	610	0.5314449414477093	{"0":0.4685550586,"1":0.5314449414}
8	0	486	0.11714112722057633	{"0":0.8828588728,"1":0.1171411272}
9	0	492	0.1258877124009584	{"0":0.8741122876,"1":0.1258877124}
10	0	489	0.12060969220628287	{"0":0.8793903078,"1":0.1206096922}
11	1	633	0.6240071289996736	{"0":0.3759928710,"1":0.6240071290}
12	0	590	0.43668648320511594	{"0":0.5633135168,"1":0.4366864832}
13	0	524	0.20197025563113366	{"0":0.7980297444,"1":0.2019702556}

Conclusions

You can use the credit card billing statements of your clients to train a scorecard model to calculate credit scores for all the clients. The credit scores can be used in loans or other credit dependent

financial transactions for assessment.

Implement image classification by TensorFlow

Overview

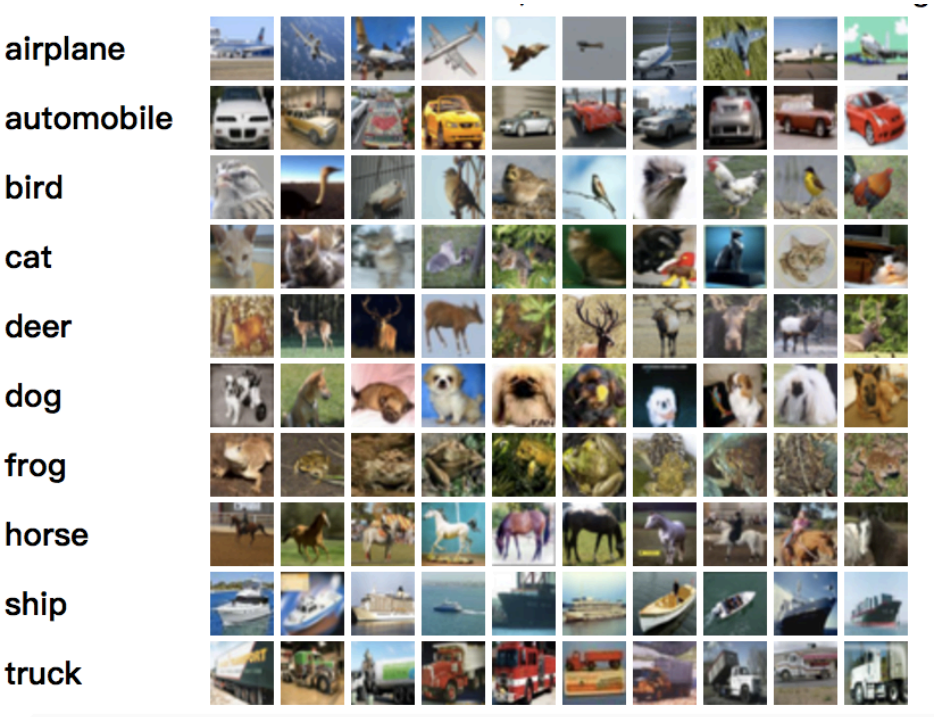
The development of the Internet has generated large volumes of images and voice data. How to effectively make use of this unstructured data has always been a challenge for data mining professionals. The processing of unstructured data usually involves the use of deep learning algorithms. These algorithms can be daunting to use at first sight. In addition, processing this data usually requires powerful GPUs and a large amount of computing resources. This document introduces a method of image recognition using deep learning frameworks. This method can be applied to scenarios such as illicit image filtering, facial recognition, and object detection.

This guide creates an image recognition model using the deep learning framework TensorFlow in Alibaba Cloud Machine Learning Platform for AI. The entire procedure takes about 30 minutes to complete. After the procedure, the system is able to recognize the bird in the following image.

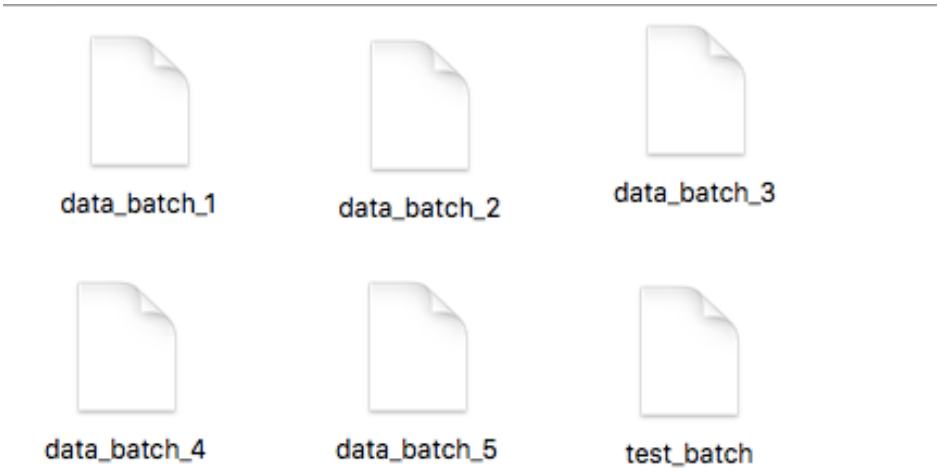
Dataset

To download the dataset and source code, click Tensorflow_cifar10 case.

The CIFAR-10 dataset is used in this guide. This dataset contains 60,000 32x32 color images in 10 different categories, such as airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. The dataset is as follows.



This source data is divided into two parts: 50,000 images are used for training and 10,000 for testing. The 50,000 training images are further divided into five data_batch files, and the 10,000 testing images form a test_batch file. The source data contains the following.



Training procedure

To create an experiment in the machine learning platform, you need to enable GPU usage and activate Object Storage Service (OSS) to store your data.

For more information about the machine learning platform, see machine learning platform console.

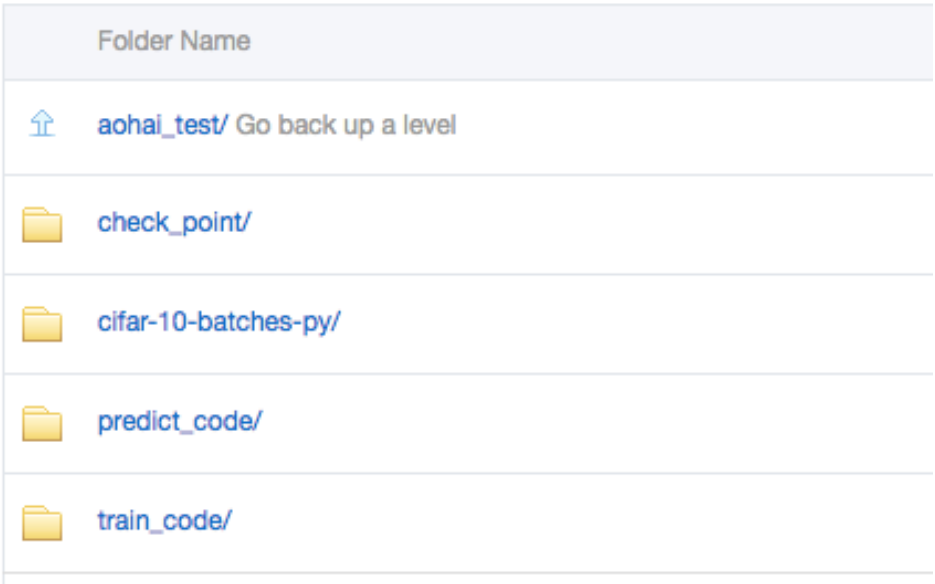
For more information about OSS, see OSS console.

1. Data preparation

Download the dataset and source code, then decompress them.

Log on to OSS, and create an OSS bucket (For more information, see OSS Document).

Create new directory in OSS bucket. An **aohai_test** directory is created in this article, and four folders are created under this directory as follows.



The role of each folder is as follows:

check_point: Stores the models that are generated in the experiment.

cifar-10-batches-py: Stores the training data, **file cifar-10-batcher-py**. The prediction data, **file bird_mount_bluebird.jpg**.

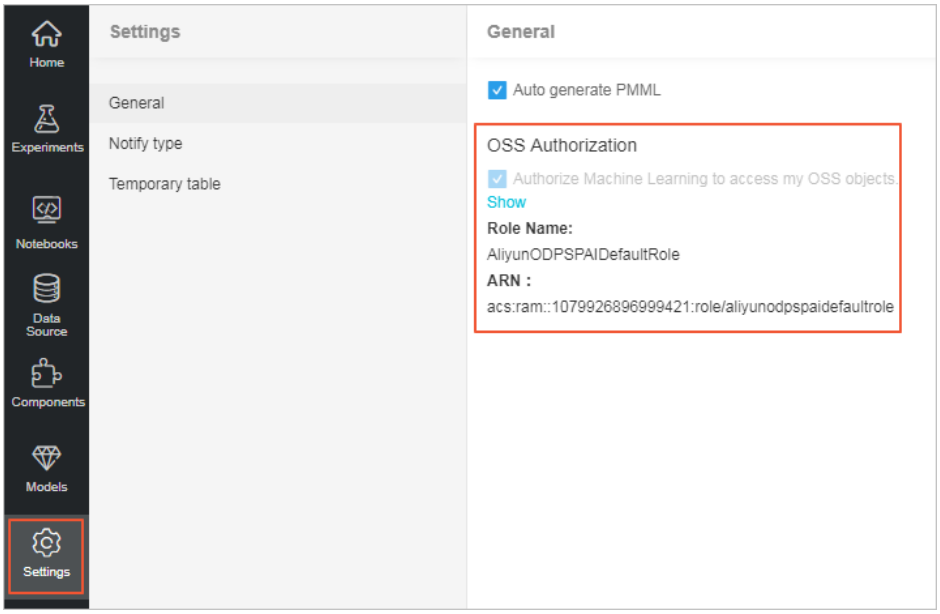
predict_code: Stores the code file **cifar_predict_pai.py**.

train_code: Stores the code **file cifar_pai.py**.

Upload the dataset and source code to the corresponding directory of the OSS bucket.

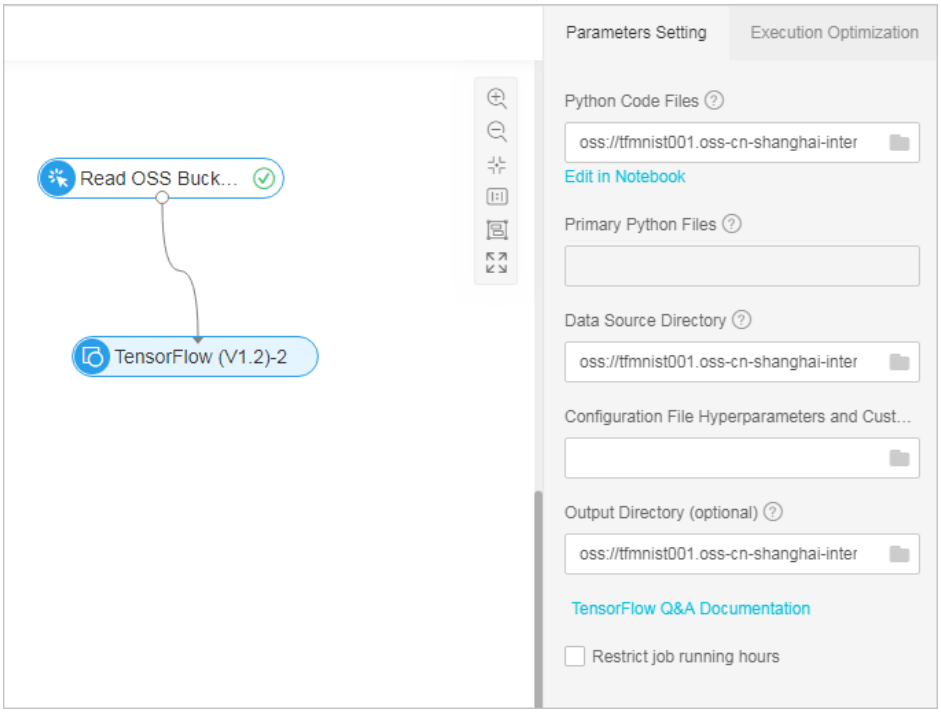
2. OSS permissions Configuration

Log on to the machine learning platform, and click **Settings** to configure OSS permissions, as shown in the following figure. For more information, see the "Read OSS buckets" chapter of Deep learning .



3. Model training

Drag a **Read OSS Bucket** component and a **TensorFlow** component to the canvas, and configure the TensorFlow component as follows.

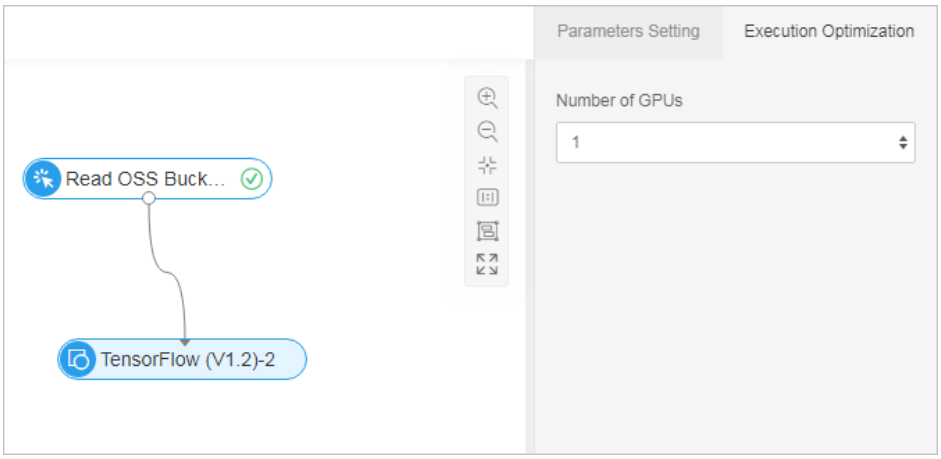


- Python Code File: Select the OSS directory of cifar_pai.py.
- Data Source Directory: Select the OSS directory of cifar-10-batches-py.

- Output Directory: Select the OSS directory of check_point.

Click **Run** to start the training procedure.

You can change the number of GPUs by changing the configuration as follows. You can also adjust the number of GPUs in the code.



4. Training code explanation

Note the following code in **cifar_pai.py**:

- The following code creates the training model using the convolutional neural network (CNN).

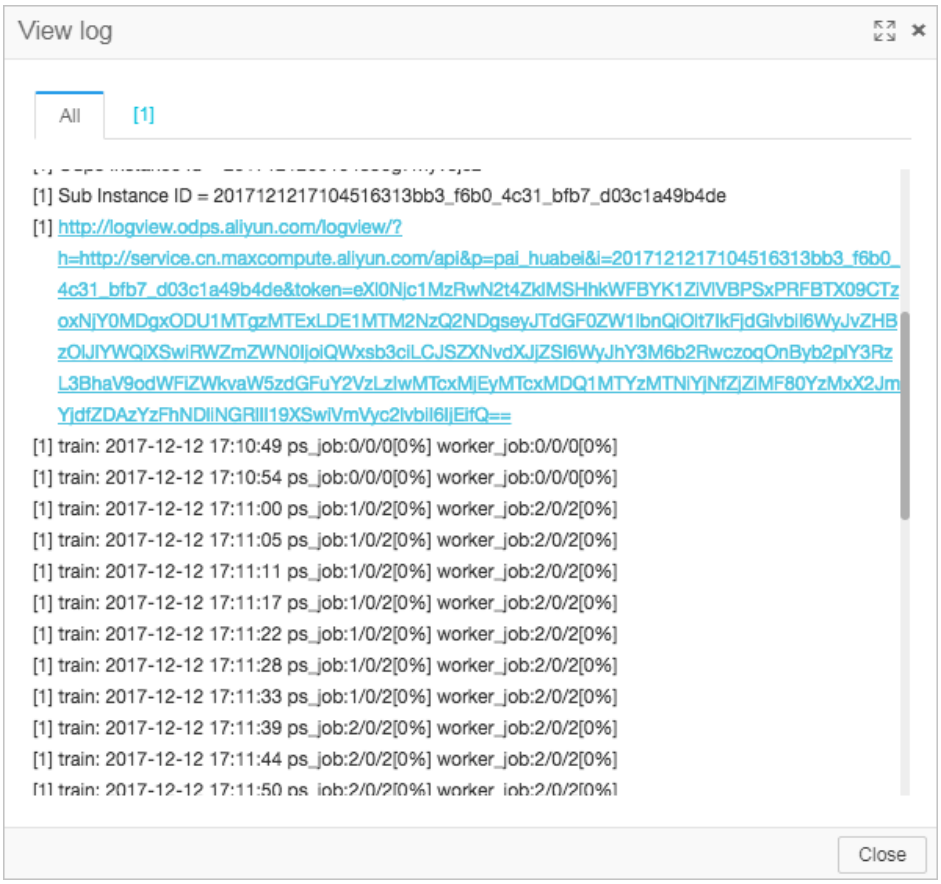
```
    network = input_data(shape=[None, 32, 32, 3],
data_preprocessing=img_prep,
data_augmentation=img_aug)
network = conv_2d(network, 32, 3, activation='relu')
network = max_pool_2d(network, 2)
network = conv_2d(network, 64, 3, activation='relu')
network = conv_2d(network, 64, 3, activation='relu')
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)
```

- The following code generates the model **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)
```

5. Log view

Right-click the TensorFlow component to view the logs generated during the training process.

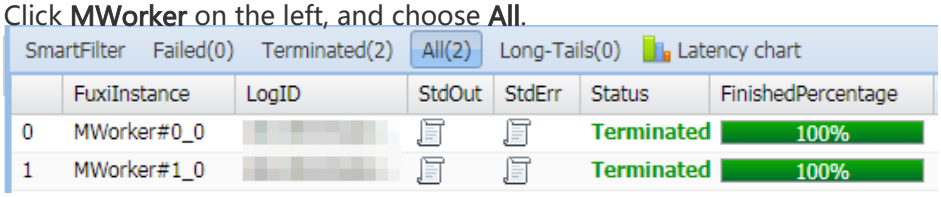


Click a logview link and run the following steps to view the logs.

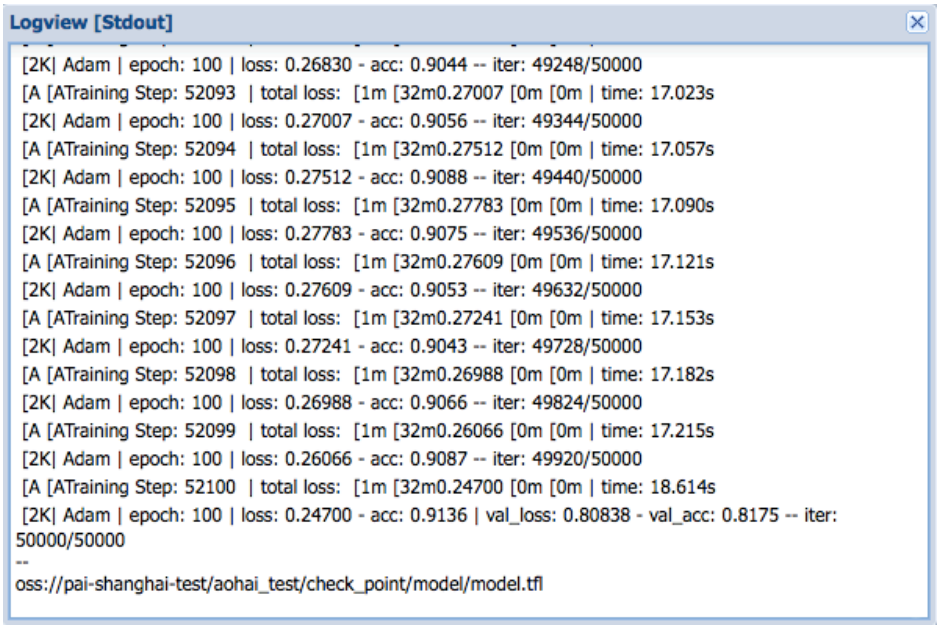
Open the **Algo Task** under **ODPS** Tasks.

Double-click the **TensorFlow Task**.

Click **MWorker** on the left, and choose **All**.



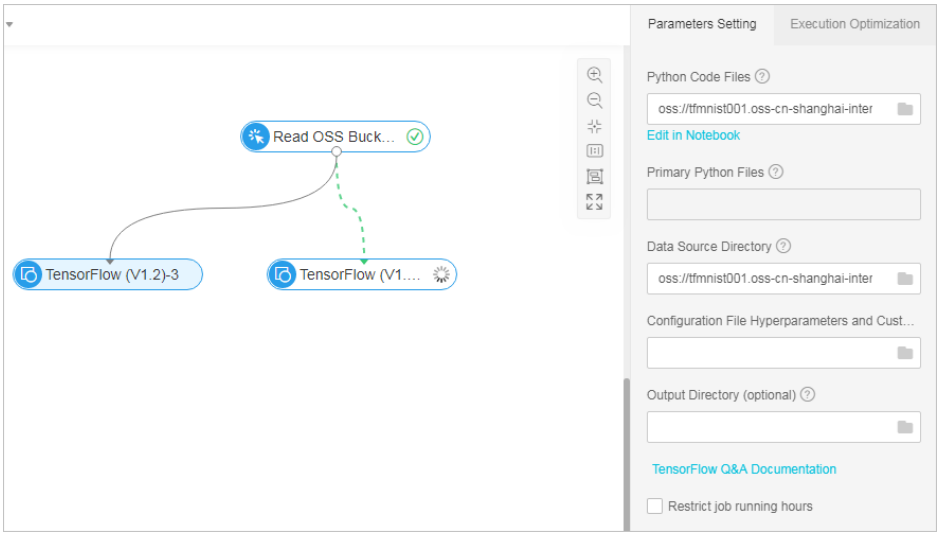
Click **StdOut** to print the training logs.



More logs are printed as the experiment continues. You can also use the print function to print key information in the code. In this example, you can use the **aac** parameter to view the accuracy of the model.

6. Result prediction

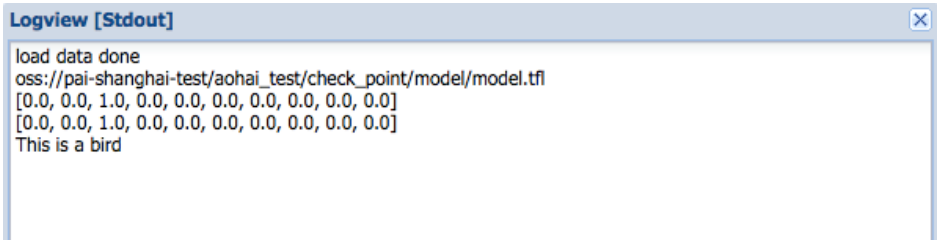
You can drag another **TensorFlow** component for use in predicting.



- Python Code File: Select the OSS directory of cifar_predict_pai.py.
- Data Source Directory: Select the OSS directory of cifar-10-batches-py.
- Output Directory: Select the OSS directory of model model.tfl.

The image that is used for predicting is stored in the **checkpoint** folder.

The prediction result is as follows:



7. Predicting code explanation

The following code:

```
  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]))]))
```

- Reads the image “bird_bullocks_oriole.jpg”, and scales the image to 32*32 pixels.
- Passes the image to the function model.predict to evaluate similarity scores.
- Returns the result based on the similarity scores. The class that scores the highest similarity is returned.

Note: Because of the randomness of the model training, it is not guaranteed that the model from each training can return accurate results for the predicted image. It is necessary to continuously debug the corresponding parameters to achieve a stable effect. This case is relatively simple and is for reference only.

Related download

Tensorflow_cifar10 case

Training data

Training code

Predicting code

Predicting image