法律声明

- 本课件包括演示文稿、示例、代码、题库、视频和声音等内容，小象学院和主讲老师拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意及内容，我们保留一切通过法律手段追究违反者的权利。

- 课程详情请咨询
  - 微信公众号：小象
  - 新浪微博：ChinaHadoop

提升实践

邹博

主要内容

- XGBoost简介
- Kaggle简介
- 代码实践

XGBoost

- XGBoost是使用梯度提升框架实现的高效、灵活、可移植的机器学习库，全称是eXtreme Gradient Boosting，是GBDT(GBM)的一个C++实现。它将树的生成并行完成，从而提高学习速度。
- 一般地说，XGBoost的速度和性能优于sklearn.ensemble.GradientBoostingClassifier类。
- XGBoost的作者为华盛顿大学陈天奇，并封装了Python接口，随着在机器学习竞赛中的优异表现，其他学者封装完成了R/Julia等接口。

XGBoost官网

- 官网：
  - https://xgboost.readthedocs.io/en/latest/
- 代码：
  - https://github.com/dmlc/xgboost/

Flexible

Supports regression, classification, ranking and user defined objectives.

Portable

Runs on Windows, Linux and OS X, as well as various cloud Platforms

Multiple Lanuages

Supports multiple languages including C++, Python, R, Java, Scala, Julia.

Battle-tested

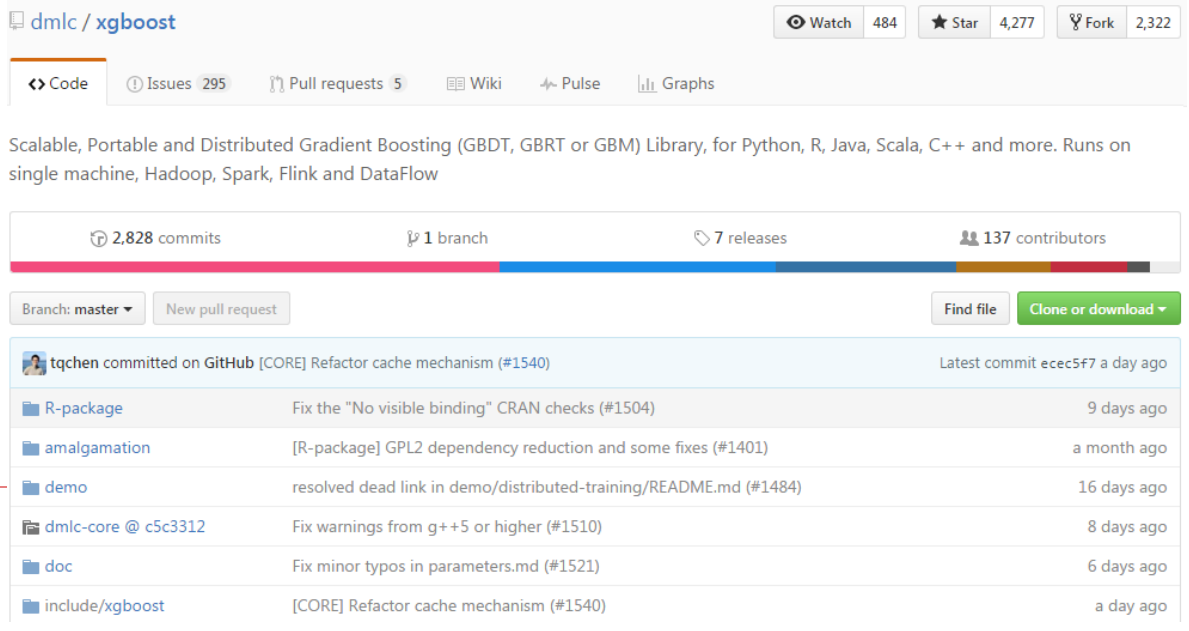
Wins many data science and machine learning challenges. Used in production by multiple companies.

Distributed on Cloud

Supports distributed training on multiple machines, including AWS, GCE, Azure, and Yarn clusters. Can be integrated with Flink, Spark and other cloud dataflow systems.

Performance

The well-optimized backend system for the best performance with limited resources. The distributed version solves problems beyond billions of examples with same code.



数据

class `xgboost. DMatrix` **(*data*, *label=None*, *missing=None*, *weight=None*, *silent=False*, *feature_names=None*, *feature_types=None*)**

Bases: `object`

Data Matrix used in XGBoost.

DMatrix is a internal data structure that used by XGBoost which is optimized for both memory efficiency and training speed. You can construct DMatrix from numpy.arrays

`feature_names`

Get feature names (column labels).

Returns: **feature_names**
Return type: list or None

`feature_types`

Get feature types (column types).

Returns: **feature_types**
Return type: list or None

训练

`xgboost.train` (*params, dtrain, num_boost_round=10, evals=(), obj=None, feval=None, maximize=False, early_stopping_rounds=None, evals_result=None, verbose_eval=True, learning_rates=None, xgb_model=None, callbacks=None*)

Train a booster with given parameters.

Parameters:

- **params** (*dict*) – Booster params.
- **dtrain** (*DMatrix*) – Data to be trained.
- **num_boost_round** (*int*) – Number of boosting iterations.
- **evals** (*list of pairs (DMatrix, string)*) – List of items to be evaluated during training, this allows user to watch performance on the validation set.
- **obj** (*function*) – Customized objective function.
- **feval** (*function*) – Customized evaluation function.
- **maximize** (*bool*) – Whether to maximize feval.
- **early_stopping_rounds** (*int*) – Activates early stopping. Validation error needs to decrease at least every <early_stopping_rounds> round(s) to continue training. Requires at least one item in evals. If there's more than one, will use the last. Returns the model from the last iteration (not the best one). If early stopping occurs, the model will have three additional fields: bst.best_score, bst.best_iteration and bst.best_ntree_limit. (Use bst.best_ntree_limit to get the correct value if num_parallel_tree and/or num_class appears in the parameters)
- **evals_result** (*dict*) –
  This dictionary stores the evaluation results of all the items in watchlist. Example: with a watchlist containing [(dtest,'eval'), (dtrain,'train')] and a paramater containing ('eval_metric', 'logloss') Returns: {'train': {'logloss': ['0.48253', '0.35953']},

  > 'eval': {'logloss': ['0.480385', '0.357756']}}

- **verbose_eval** (*bool or int*) – Requires at least one item in evals. If *verbose_eval* is True then the evaluation metric on the validation set is printed at each boosting stage. If *verbose_eval* is an integer then the evaluation metric on the validation set is printed at every given *verbose_eval* boosting stage. The last boosting stage / the boosting stage found by using *early_stopping_rounds* is also printed. Example: with verbose_eval=4 and at least one item in evals, an evaluation metric is printed every 4 boosting stages, instead of every boosting stage.
- **learning_rates** (*list or function*) – List of learning rate for each boosting round or a customized function that calculates eta in terms of current number of round and the total number of boosting round (e.g. yields learning rate decay) - list l: eta = l[boosting round] - function f: eta = f(boosting round, num_boost_round)
- **xgb_model** (*file name of stored xgb model or 'Booster' instance*) – Xgb model to be loaded before training (allows training continuation).
- **callbacks** (*list of callback functions*) – List of callback functions that are applied at end of each iteration.

Returns: **booster**

Return a trained booster model

预测

predict (*data*, *output_margin=False*, *ntree_limit=0*, *pred_leaf=False*)

Predict with data.

NOTE: This function is not thread safe.

For each booster object, predict can only be called from one thread. If you want to run prediction using multiple thread, call bst.copy() to make copies of model object and then call predict

Parameters:

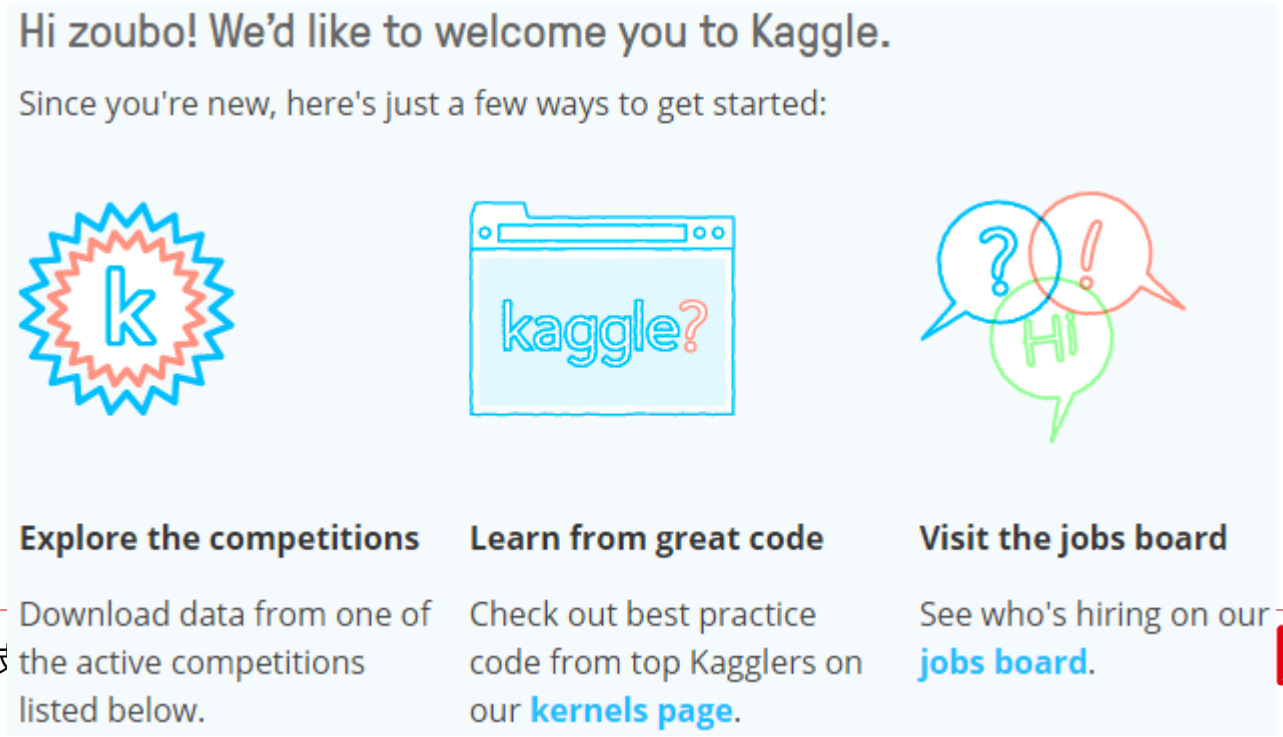
- **data** (*DMatrix*) – The dmatrix storing the input.
- **output_margin** (*bool*) – Whether to output the raw untransformed margin value.
- **ntree_limit** (*int*) – Limit number of trees in the prediction; defaults to 0 (use all trees).
- **pred_leaf** (*bool*) – When this option is on, the output will be a matrix of (nsample, ntrees) with each record indicating the predicted leaf index of each sample in each tree. Note that the leaf index of a tree is unique per tree, so you may find leaf 1 in both tree 1 and tree 0.

Returns: **prediction**

Return numpy array

Kaggle简介

- Kaggle是一个数据分析的竞赛平台，网址：https://www.kaggle.com/。
- 注册新账号后的导航界面：



Kaggle类别

Meta Kaggle
Kaggle's public data on competitions, users, submission scores, and kernels

Amazon Fine Food Reviews
Analyze ~500,000 food reviews from Amazon

NBA shot logs
Moneyball data, for basketball.

101

Digit Recognizer
Classify handwritten digits using the famous MNIST data

Titanic: Machine Learning from Disaster
Predict survival on the Titanic using Excel, Python, R & Random Forests

Facial Keypoints Detection
Detect the location of keypoints on face images

First Steps With Julia
Use Julia to identify characters from Google Street View images

Active Competitions

Predicting Red Hat Business Value
Classify customer potential
15 days
1685 teams
1470 kernels

Bosch Production Line Performance
Reduce manufacturing failures
2 months
224 teams
$30,000

TalkingData Mobile User Demographics
Get to know millions of mobile device users
41 hours
1714 teams
2813 kernels

Melbourne University AES/MathWorks/NIH Seiz...
Predict seizures in long-term human intracranial EEG recordings
2 months
46 teams
$20,000

Integer Sequence Learning
1, 2, 3, 4, 5, 7?!
26 days
218 teams
415 kernels
Knowledge

Painter by Numbers
Does every painter leave a fingerprint?
57 days
29 teams
92 kernels
Knowledge

Leaf Classification
Can you see the random forest for the leaves?
5 months
52 teams
71 kernels
Knowledge

House Prices: Advanced Regression Techniques
Sold! How do home features add up to its price tag?
5 months
83 teams
80 kernels
Knowledge

Dogs vs. Cats Redux: Kernels Edition
Distinguish images of dogs from cats
5 months
3 teams
4 kernels
Knowledge

Knowledge • 4,690 teams

Titanic: Machine Learning from Disaster

Fri 28 Sep 2012 — Sat 31 Dec 2016 (3 months to go)

Predict survival on the Titanic using Excel, Python, R & Random Forests

If you're new to data science and machine learning, or looking for a simple intro to the Kaggle competitions platform, this is the best place to start. Continue reading below the competition description to discover a number of tutorials, benchmark models, and more.

Competition Description

The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.

One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class.

In this challenge, we ask you to complete the analysis of what sorts of people were likely to survive. In particular, we ask you to apply the tools of machine learning to predict which passengers survived the tragedy.

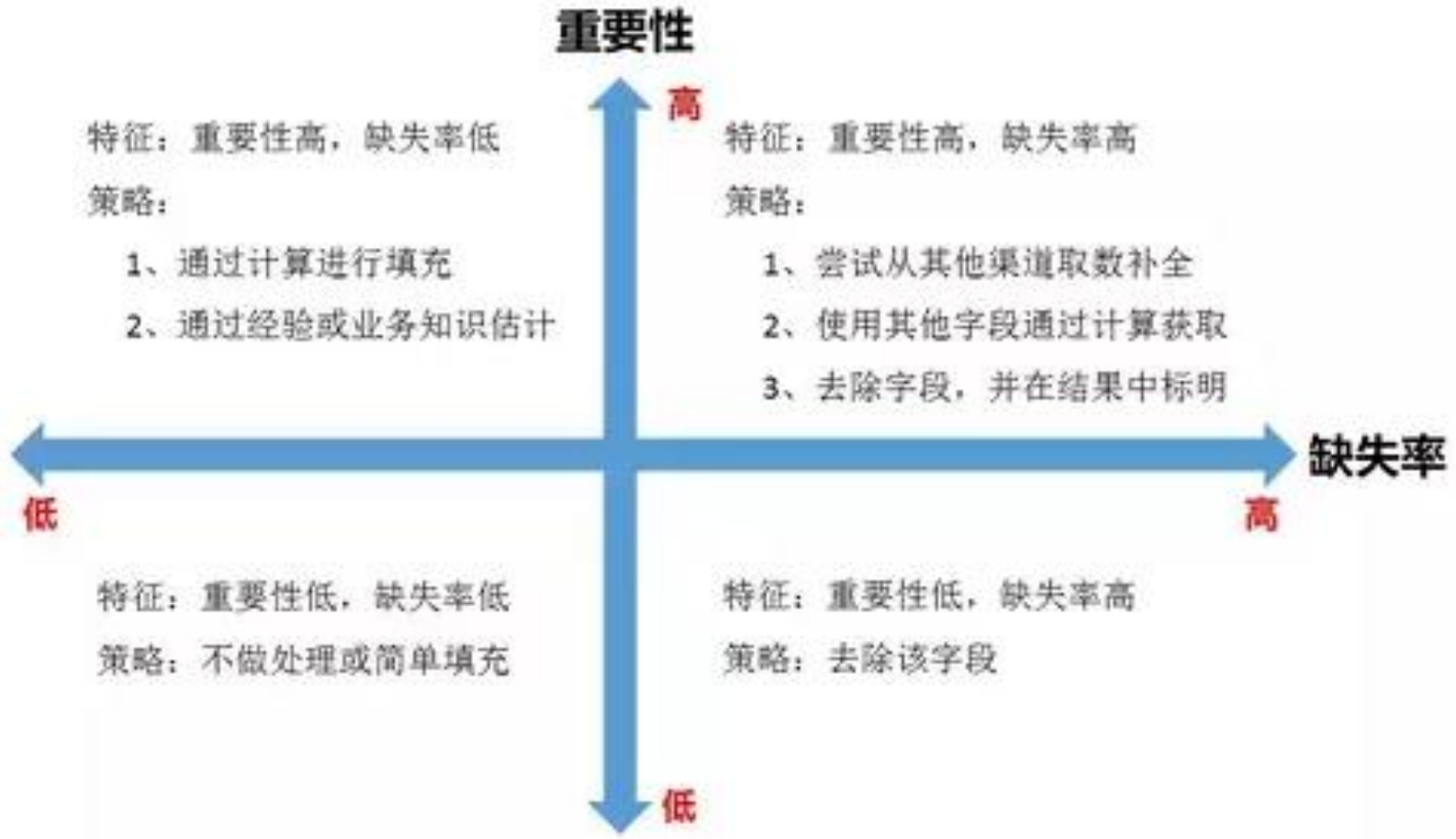
数据

A	B	C	D	E	F	G	H	I	J	K	L
PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.25		S
2	1	1	John Bradley (Florence Br	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26	0	0	ON/O2. 31012	7.925		S
4	1	1	Mrs. Jacques Heath (Lily	female	35	1	0	113803	53.1	C123	S
5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.05		S
6	0	3	Moran, Mr. James	male		0	0	330877	8.4583		Q
7	0	1	McCarthy, Mr. Timothy J	male	54	0	0	17463	51.8625	E46	S
8	0	3	lsson, Master. Gosta Leona	male	2	3	1	349909	21.075		S
9	1	3	. Oscar W (Elisabeth Vilhe	female	27	0	2	347742	11.1333		S
10	1	2	r, Mrs. Nicholas (Adele Ac	female	14	1	0	237736	30.0708		C
11	1	3	dstrom, Miss. Marguerite I	female	4	1	1	PP 9549	16.7	G6	S
12	1	1	Bonnell, Miss. Elizabeth	female	58	0	0	113783	26.55	C103	S
13	0	3	undercock, Mr. William Hen	male	20	0	0	A/5. 2151	8.05		S
14	0	3	ndersson, Mr. Anders Johar	male	39	1	5	347082	31.275		S
15	0	3	om, Miss. Hulda Amanda Ado	female	14	0	0	350406	7.8542		S
16	1	2	lett, Mrs. (Mary D Kingcom	female	55	0	0	248706	16		S
17	0	3	Rice, Master. Eugene	male	2	4	1	382652	29.125		Q
18	1	2	illiams, Mr. Charles Eugen	male		0	0	244373	13		S
19	0	3	Mrs. Julius (Emelia Maria	female	31	1	0	345763	18		S
20	1	3	Masselmani, Mrs. Fatima	female		0	0	2649	7.225		C
21	0	2	Fynney, Mr. Joseph J	male	35	0	0	239865	26		S
22	1	2	Beesley, Mr. Lawrence	male	34	0	0	248698	13	D56	S
23	1	3	lcGowan, Miss. Anna "Annie'	female	15	0	0	330923	8.0292		Q
24	1	1	loper, Mr. William Thompso	male	28	0	0	113788	35.5	A6	S
25	0	3	lsson, Miss. Torborg Danir	female	8	3	1	349909	21.075		S
26	1	3	rl Oscar (Selma Augusta Er	female	38	1	5	347077	31.3875		S
27	0	3	Emir, Mr. Farred Chehab	male		0	0	2631	7.225		C
28	0	1	rtune, Mr. Charles Alexand	male	19	3	2	19950	263	C23 C25 C27	S
29	1	3	Dwyer, Miss. Ellen "Nellie	female		0	0	330959	7.8792		Q
30	0	3	Todoroff, Mr. Lalio	male		0	0	349216	7.8958		S
31	0	1	Uruchurtu, Don. Manuel E	male	40	0	0	PC 17601	27.7208		C
32	1	1	rs. William Augustus (Mari	female		1	0	PC 17569	146.5208	B78	C
33	1	3	Glynn, Miss. Mary Agatha	female		0	0	335677	7.75		Q

数据说明

```
VARIABLE DESCRIPTIONS:
survival        Survival
                (0 = No; 1 = Yes)
pclass          Passenger Class
                (1 = 1st; 2 = 2nd; 3 = 3rd)
name            Name
sex             Sex
age             Age
sibsp           Number of Siblings/Spouses Aboard
parch           Number of Parents/Children Aboard
ticket          Ticket Number
fare            Passenger Fare
cabin           Cabin
embarked        Port of Embarkation
                (C = Cherbourg; Q = Queenstown; S = Southampton)

SPECIAL NOTES:
Pclass is a proxy for socio-economic status (SES)
 1st ~ Upper; 2nd ~ Middle; 3rd ~ Lower

Age is in Years; Fractional if Age less than One (1)
 If the Age is Estimated, it is in the form xx.5

With respect to the family relation variables (i.e. sibsp and parch)
some relations were ignored.  The following are the definitions used
for sibsp and parch.

Sibling:  Brother, Sister, Stepbrother, or Stepsister of Passenger Aboard Titanic
Spouse:   Husband or Wife of Passenger Aboard Titanic (Mistresses and Fiances Ignored)
Parent:   Mother or Father of Passenger Aboard Titanic
Child:    Son, Daughter, Stepson, or Stepdaughter of Passenger Aboard Titanic

Other family relatives excluded from this study include cousins,
nephews/nieces, aunts/uncles, and in-laws.  Some children travelled
only with a nanny, therefore parch=0 for them.  As well, some
travelled with very close friends or neighbors in a village, however,
the definitions do not support such relations.
```

数据预处理 – 清洗


重要性
高
特征：重要性高，缺失率低
策略：
1、通过计算进行填充
2、通过经验或业务知识估计
特征：重要性高，缺失率高
策略：
1、尝试从其他渠道取数补全
2、使用其他字段通过计算获取
3、去除字段，并在结果中标明
缺失率
低
高
特征：重要性低，缺失率低
策略：不做处理或简单填充
特征：重要性低，缺失率高
策略：去除该字段
低

数据处理

```
def load_data(file_name, is_train):
    data = pd.read_csv(file_name)  # 数据文件路径
    # print data.describe()

    # 性别
    data['Sex'] = data['Sex'].map({'female': 0, 'male': 1}).astype(int)

    # 补齐船票价格缺失值
    if len(data.Fare[data.Fare.isnull()]) > 0:
        fare = np.zeros(3)
        for f in range(0, 3):
            fare[f] = data[data.Pclass == f + 1]['Fare'].dropna().median()
        for f in range(0, 3):  # loop 0 to 2
            data.loc[(data.Fare.isnull()) & (data.Pclass == f + 1), 'Fare'] = fare[f]

    # 年龄：使用均值代替缺失值
    # mean_age = data['Age'].dropna().mean()
    # data.loc[(data.Age.isnull()), 'Age'] = mean_age
    if is_train:
        # 年龄：使用随机森林预测年龄缺失值
        print '随机森林预测缺失年龄：--start--'
        data_for_age = data[['Age', 'Survived', 'Fare', 'Parch', 'SibSp', 'Pclass']]
        age_exist = data_for_age.loc[(data.Age.notnull())]   # 年龄不缺失的数据
        age_null = data_for_age.loc[(data.Age.isnull())]
        # print age_exist
        x = age_exist.values[:, 1:]
        y = age_exist.values[:, 0]
        rfr = RandomForestRegressor(n_estimators=1000)
        rfr.fit(x, y)
        age_hat = rfr.predict(age_null.values[:, 1:])
        # print age_hat
        data.loc[(data.Age.isnull()), 'Age'] = age_hat
        print '随机森林预测缺失年龄：--over--'
```

预测

```python
if __name__ == "__main__":
    x, y = load_data('8.Titanic.train.csv', True)
    x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.5, random_s

    lr = LogisticRegression(penalty='l2')
    lr.fit(x_train, y_train)
    y_hat = lr.predict(x_test)
    lr_rate = show_accuracy(y_hat, y_test, 'Logistic回归 ')
    # write_result(lr, 1)

    rfc = RandomForestClassifier(n_estimators=100)
    rfc.fit(x_train, y_train)
    y_hat = rfc.predict(x_test)
    rfc_rate = show_accuracy(y_hat, y_test, '随机森林 ')
    # write_result(rfc, 2)

    # XGBoost
    data_train = xgb.DMatrix(x_train, label=y_train)
    data_test = xgb.DMatrix(x_test, label=y_test)
    watch_list = [(data_test, 'eval'), (data_train, 'train')]
    param = {'max_depth': 3, 'eta': 0.1, 'silent': 1, 'objective': 'binary:logistic'}
            # 'subsample': 1, 'alpha': 0, 'lambda': 0, 'min_child_weight': 1}
    bst = xgb.train(param, data_train, num_boost_round=100, evals=watch_list)
    y_hat = bst.predict(data_test)
```

8.5.Titanic

```
[92]    eval-error:0.143605 train-error:0.103293
[93]    eval-error:0.143605 train-error:0.103293
[94]    eval-error:0.143605 train-error:0.103293
[95]    eval-error:0.144353 train-error:0.104790
[96]    eval-error:0.144353 train-error:0.104790
[97]    eval-error:0.144353 train-error:0.104790
[98]    eval-error:0.146597 train-error:0.104790
[99]    eval-error:0.146597 train-error:0.104790
Logistic回归：78.833%
随机森林：92.745%
XGBoost：85.340%
```

Wine数据集

- 本数据来自于意大利某地区的葡萄酒数据，使用该地区3种不同品种的葡萄酿制葡萄酒，发现葡萄酒的化学分析成分是不同的，数据集共178个样本，各类别数目为59、71、48， 13个特征包括：
  - Alcohol(酒精度)、Malic acid(苹果酸)、Ash(灰成分)、Alcalinity of ash(灰分的碱性值)、Magnesium(镁含量)、Total phenols(苯酚总量)、Flavanoids(黄酮)、Nonflavanoid phenols(非黄酮类苯酚总量)、Proanthocyanins(原花青素)、Color intensity(颜色强度)、Hue(色调)、OD280/OD315 of diluted wines(280nm/315nm吸光度)、Proline(脯氨酸)
  - 下载地址：http://archive.ics.uci.edu/ml/machine-learning-databases/wine/

Wine数据集

- 酒精度
- 苹果酸
- 灰分
- 灰分碱性值
- 镁含量
- 苯酚总量
- 黄酮
- 非黄酮类苯酚总量
- 原花青素
- 颜色强度
- 色调
- 280/315nm吸光度
- 脯氨酸

1,14.23,1.71,2.43,15.6,127,2.8,3.06,.28,2.29,5.64,1.04,3.92,1065
1,13.2,1.78,2.14,11.2,100,2.65,2.76,.26,1.28,4.38,1.05,3.4,1050
1,13.16,2.36,2.67,18.6,101,2.8,3.24,.3,2.81,5.68,1.03,3.17,1185
1,14.37,1.95,2.5,16.8,113,3.85,3.49,.24,2.18,7.8,.86,3.45,1480
1,13.24,2.59,2.87,21,118,2.8,2.69,.39,1.82,4.32,1.04,2.93,735
1,14.2,1.76,2.45,15.2,112,3.27,3.39,.34,1.97,6.75,1.05,2.85,1450
1,14.39,1.87,2.45,14.6,96,2.5,2.52,.3,1.98,5.25,1.02,3.58,1290
1,14.06,2.15,2.61,17.6,121,2.6,2.51,.31,1.25,5.05,1.06,3.58,1295
1,14.83,1.64,2.17,14,97,2.8,2.98,.29,1.98,5.2,1.08,2.85,1045
1,13.86,1.35,2.27,16,98,2.98,3.15,.22,1.85,7.22,1.01,3.55,1045
1,14.1,2.16,2.3,18,105,2.95,3.32,.22,2.38,5.75,1.25,3.17,1510
1,14.12,1.48,2.32,16.8,95,2.2,2.43,.26,1.57,5,1.17,2.82,1280
1,13.75,1.73,2.41,16,89,2.6,2.76,.29,1.81,5.6,1.15,2.9,1320
1,14.75,1.73,2.39,11.4,91,3.1,3.69,.43,2.81,5.4,1.25,2.73,1150
1,14.38,1.87,2.38,12,102,3.3,3.64,.29,2.96,7.5,1.2,3,1547
1,13.63,1.81,2.7,17.2,112,2.85,2.91,.3,1.46,7.3,1.28,2.88,1310
1,14.3,1.92,2.72,20,120,2.8,3.14,.33,1.97,6.2,1.07,2.65,1280
1,13.83,1.57,2.62,20,115,2.95,3.4,.4,1.72,6.6,1.13,2.57,1130
1,14.19,1.59,2.48,16.5,108,3.3,3.93,.32,1.86,8.7,1.23,2.82,1680
1,13.64,3.1,2.56,15.2,116,2.7,3.03,.17,1.66,5.1,.96,3.36,845
1,14.06,1.63,2.28,16,126,3,3.17,.24,2.1,5.65,1.09,3.71,780
1,12.93,3.8,2.65,18.6,102,2.41,2.41,.25,1.98,4.5,1.03,3.52,770
1,13.71,1.86,2.36,16.6,101,2.61,2.88,.27,1.69,3.8,1.11,4,1035
1,12.85,1.6,2.52,17.8,95,2.48,2.37,.26,1.46,3.93,1.09,3.63,1015
1,13.5,1.81,2.61,20,96,2.53,2.61,.28,1.66,3.52,1.12,3.82,845
1,13.05,2.05,3.22,25,124,2.63,2.68,.47,1.92,3.58,1.13,3.2,830
1,13.39,1.77,2.62,16.1,93,2.85,2.94,.34,1.45,4.8,.92,3.22,1195
1,13.3,1.72,2.14,17,94,2.4,2.19,.27,1.35,3.95,1.02,2.77,1285

Code

```
if __name__ == "__main__":
    data = pd.read_csv('wine.data', header=None)
    x, y = data.iloc[:, 1:], data[0]
    x = MinMaxScaler().fit_transform(x)
    x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=1, test_size=0.7)

    lr = RidgeClassifierCV(alphas=np.logspace(-3, 3, 10), cv=3)
    lr.fit(x_train, y_train.ravel())
    print u'参数alpha=%.2f' % lr.alpha_
    y_train_pred = lr.predict(x_train)
    y_test_pred = lr.predict(x_test)
    print u'Logistic回归训练集准确率：', accuracy_score(y_train, y_train_pred)
    print u'Logistic回归测试集准确率：', accuracy_score(y_test, y_test_pred)

    rf = RandomForestClassifier(n_estimators=100, max_depth=8, min_samples_split=5, oob_score=True)
    rf.fit(x_train, y_train.ravel())
    print u'OOB Score=%.5f' % rf.oob_score_
    y_train_pred = rf.predict(x_train)
    y_test_pred = rf.predict(x_test)
    print u'随机森林训练集准确率：', accuracy_score(y_train, y_train_pred)
    print u'随机森林测试集准确率：', accuracy_score(y_test, y_test_pred)

    gb = GradientBoostingClassifier(n_estimators=100, learning_rate=0.1, max_depth=2)
    gb.fit(x_train, y_train.ravel())
    y_train_pred = gb.predict(x_train)
    y_test_pred = gb.predict(x_test)
    print u'GBDT训练集准确率：', accuracy_score(y_train, y_tra
    print u'GBDT测试集准确率：', accuracy_score(y_test, y_test

    y_train[y_train == 3] = 0
    y_test[y_test == 3] = 0
    data_train = xgb.DMatrix(x_train, label=y_train)
    data_test = xgb.DMatrix(x_test, label=y_test)
    watch_list = [(data_test, 'eval'), (data_train, 'train')]
    params = {'max_depth': 1, 'eta': 0.9, 'silent': 1, 'objec
    bst = xgb.train(params, data_train, num_boost_round=5, ev
    y_train_pred = bst.predict(data_train)
    y_test_pred = bst.predict(data_test)
    print u'XGBoost训练集准确率：', accuracy_score(y_train, y_train_pred)
    print u'XGBoost测试集准确率：', accuracy_score(y_test, y_test_pred)
```

```
参数alpha=0.46
Logistic回归训练集准确率： 1.0
Logistic回归测试集准确率： 0.968
OOB Score=0.94340
随机森林训练集准确率： 1.0
随机森林测试集准确率： 0.976
GBDT训练集准确率： 1.0
GBDT测试集准确率： 0.984
XGBoost训练集准确率： 1.0
XGBoost测试集准确率： 0.96
```

作业

- 安装XGBoost，并使用提供的Wine/Iris/Adult等数据集做分类预测。

参考文献

- Tianqi Chen and Carlos Guestrin. *XGBoost: A Scalable Tree Boosting System*. In 22nd SIGKDD Conference on Knowledge Discovery and Data Mining, 2016
- API：http://xgboost.readthedocs.io/en/latest/python/python_api.html
- Python：https://github.com/dmlc/xgboost/tree/master/demo/guide-python
- 介绍：https://xgboost.readthedocs.io/en/latest/model.html

我们在这里

- http://wenda.ChinaHadoop.cn
  - 视频/课程/社区
- 微博
  - @ChinaHadoop
  - @邹博_机器学习
- 微信公众号
  - 小象学院
  - 大数据分析挖掘

感谢大家！

恳请大家批评指正！