



第一讲：初识XGBoost

AI100学院
2017年6月



► Roadmap

- XGBoost简介
- XGBoost的优势
- 与XGBoost的第一次亲密接触



► XGBoost



- XGBoost : eXtreme Gradient Boosting
- Gradient Boosting Machines(GBM) 的C++优化实现，快速有效
 - 深盟分布式机器学习开源平台 (Distributed Machine Learning Community , DMLC)的一个分支
 - DMLC也开包含流行的深度学习库mxnet

The name xgboost, though, actually refers to the engineering goal to **push the limit of computations resources for boosted tree algorithms**. Which is the reason why many people use xgboost.

– Tianqi



Chen, on Quora.com



► Roadmap

- XGBoost简介
 - Gradient Boosting Machines
 - XGBoost的特别之处
- XGBoost的优势
- 与XGBoost的第一次亲密接触



► Gradient Boosting Machines

- Machines : 机器学习模型
 - 对数据的产生规律建模
- Boosting Machines
 - 弱学习器组合成强学习器 / 模型
- Gradient Boosting Machines
 - 根据梯度下降方式组合弱学习器



► Machines

- Machines：机器学习模型，建模数据产生规律
 - 最小化目标函数
- 目标函数通常包含两部分
 - **损失函数**：与任务有关（选择与训练数据匹配最好的模型）
 - 回归：残差平方
 - 分类：0-1损失、logistic损失、合叶损失（SVM）
 - ...
 - **正则项**：与模型复杂度度有关（选择最简单的模型）
 - L2正则
 - L1正则
 - ...



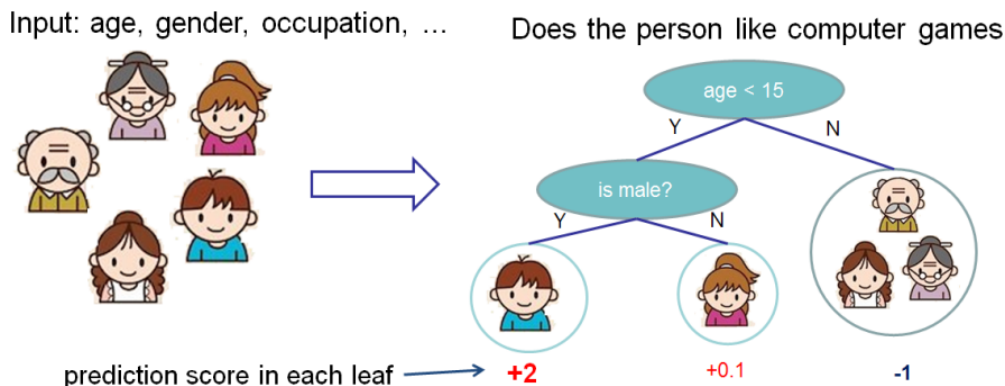
Occam剃刀原理

<http://www.ai100.ai/>



▶ Boosting Machines

- Boosting: 将弱学习器组合成强学习器
- **弱学习器**：比随机猜测性能好的学习器
- 常用弱学习器：决策树 / 分类回归树
 - 决策树：每个叶子节点对应一个决策
 - 分类回归树：每个叶子节点有个预测分数(score)，比决策树更灵活



► Boosting Machines (cont.)

- **Adaptive Boosting** (AdaBoost): 第一个Boosting算法
- 弱分类器：只有一个分裂的决策树
 - if $\text{feature_value} > t$, output 1; otherwise output 0
- 对当前分类器不能处理的样本，增加其权重
- 不断的加入新的弱学习器，直到达到终止条件
 - 强学习器：弱学习器的加权线性组合，权重与其正确率有关



三个臭皮匠，顶个诸葛亮

<http://www.ai100.ai/>



► Gradient Boosting Machines

- Friedman将AdaBoost推广到一般Gradient Boosting框架，得到Gradient Boosting Machines (GBM): 将boosting视作一个数值优化问题，采用类似**梯度下降**的方式优化求解
 - 亦被称为stage-wise additive model: 每次增加一个弱学习器到已有模型，已在模型中的弱学习器不再变化
 - 这种推广使得我们可以使用任何可微的损失函数，从而支持的任务从两类分类扩展到回归和多类分类问题。



► Roadmap

- XGBoost简介
 - Gradient Boosting Machines
 - XGBoost的特别之处
- XGBoost的优势
- 与XGBoost的第一次亲密接触



► XGBoost的特别之处

- 正则化: 以 “正则化提升(regularized boosting)” 著称
 - 标准GBM的实现没有显式的正则化步骤
 - 正则化对减少过拟合有帮助
- 并行处理，相比GBM有了速度的飞跃
 - 借助 OpenMP，自动利用单机CPU的多核进行并行计算
 - 支持GPU加速
 - 支持分布式
- 高度的灵活性：允许用户自定义优化目标和评价标准
 - 只需损失函数的一阶导数和二阶导数



► XGBoost的特别之处 (cont.)

- 剪枝
 - 当新增分裂带来负增益时，GBM会停止分裂
 - XGBoost一直分裂到指定的最大深度(max_depth)，然后回过头来剪枝
- 内置交叉验证
 - XGBoost允许在每一轮boosting迭代中使用交叉验证 → 可以方便地获得最优boosting迭代次数
 - GBM使用网格搜索，只能检测有限个值
- 在线学习：XGBoost和GBM都支持



► XGBoost的多语言接口

- 提供多语言接口
 - 命令行 (Command Line Interface , CLI)
 - C++
 - Python (可以和scikit-learn结合)
 - R (可以和caret包结合)
 - Julia
 - JAVA和JVM语言 (如Scala、Hadoop平台等)



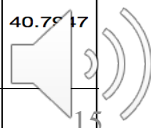
► Roadmap

- XGBoost简介
- XGBoost的优势
- 与XGBoost的第一次亲密接触
 - XGBoost 独立使用
 - 与scikit-learn一起使用

► XGBoost的优势

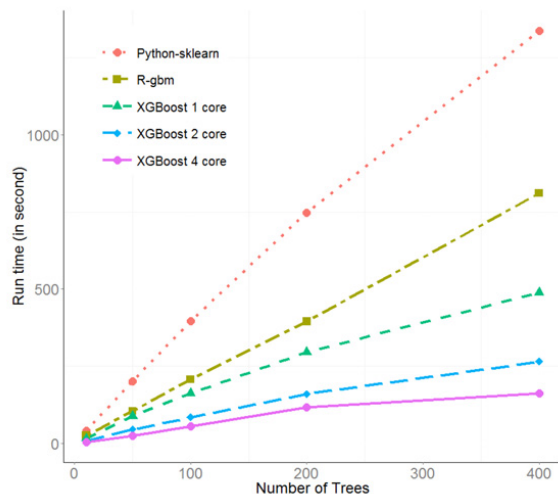
- XGBoost是近几年应用机器学习领域内一个强有力的武器
 - 执行速度：确实比其他Gradient Boosting实现快
 - 模型性能：在结构化数据集上，在分类 / 回归/排序预测建模上表现突出

	bathrooms	bedrooms	building_id	created	description	display_address	features	interest_level	latitude
10	1.5	3	53a5b119ba8f7b61d4e010512e0dfc85	2016-06-24 07:54:24	A Brand New 3 Bedroom 1.5 bath ApartmentEnjoy ...	Metropolitan Avenue	[]	medium	40.7145
10000	1.0	2	c5c8a357cba207596b04d1afd1e4f130	2016-06-12 12:19:27		Columbus Avenue	[Doorman, Elevator, Fitness Center, Cats Allow...	low	40.7147
100004	1.0	1	c3ba40552e2120b0acfc3cb5730bb2aa	2016-04-17	Top Top West Village location,	W 13 Street	[Laundry In Building, Dishwasher,	high	40.7388



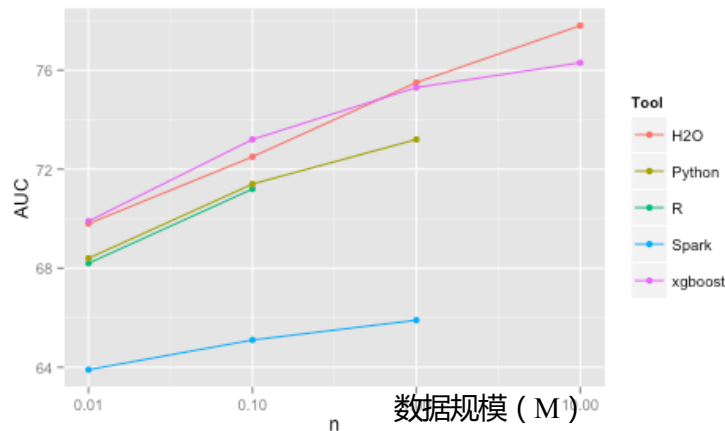
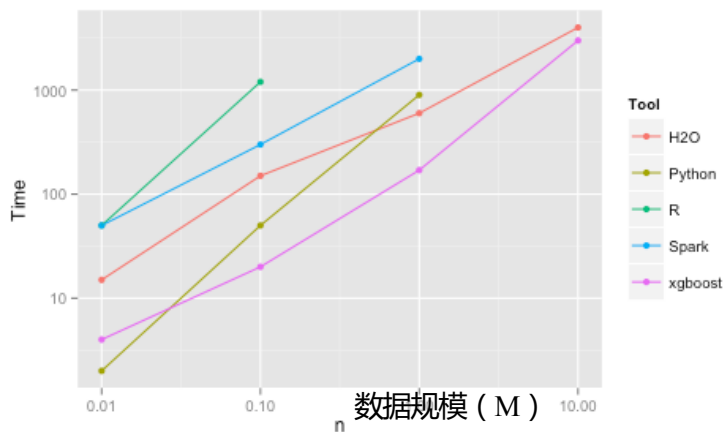
▶ 执行速度

- 在Kaggle的Higgs Boson竞赛中
 - 单线程的XGBoost比GBM的R语言包实现和Python的sklearn实现快将近一倍
 - 多线程有接近线性程度的提升



▶ 执行速度 (cont.)

- 2015年, Szilard Pafka 在一些数据集上比较了 XGBoost 和其他 Gradient Boosting / Bagged decision trees 实现：又快又好



► XGBoost的性能

- 在Kaggle和KDD等一些数据科学竞赛平台上取了优异成绩

竞赛	名次	作者	备注
KDD Cup 2016 competition.	1	Vlad Sandulescu, Mihai Chiru	http://arxiv.org/abs/1609.02728
Dato Truly Native? competition.	1	Marios Michailidis, Mathias Müller and HJ van Veen	http://blog.kaggle.com/2015/12/03/dato-winners-interview-1st-place-mad-professors/
CERN LHCb experiment Flavour of Physics competition.	1	Vlad Mironov, Alexander Guschin	http://blog.kaggle.com/2015/11/30/flavour-of-physics-technical-write-up-1st-place-go-polar-bears/
...



<https://github.com/dmlc/xgboost/tree/master>

<http://www.ai100.ai/>



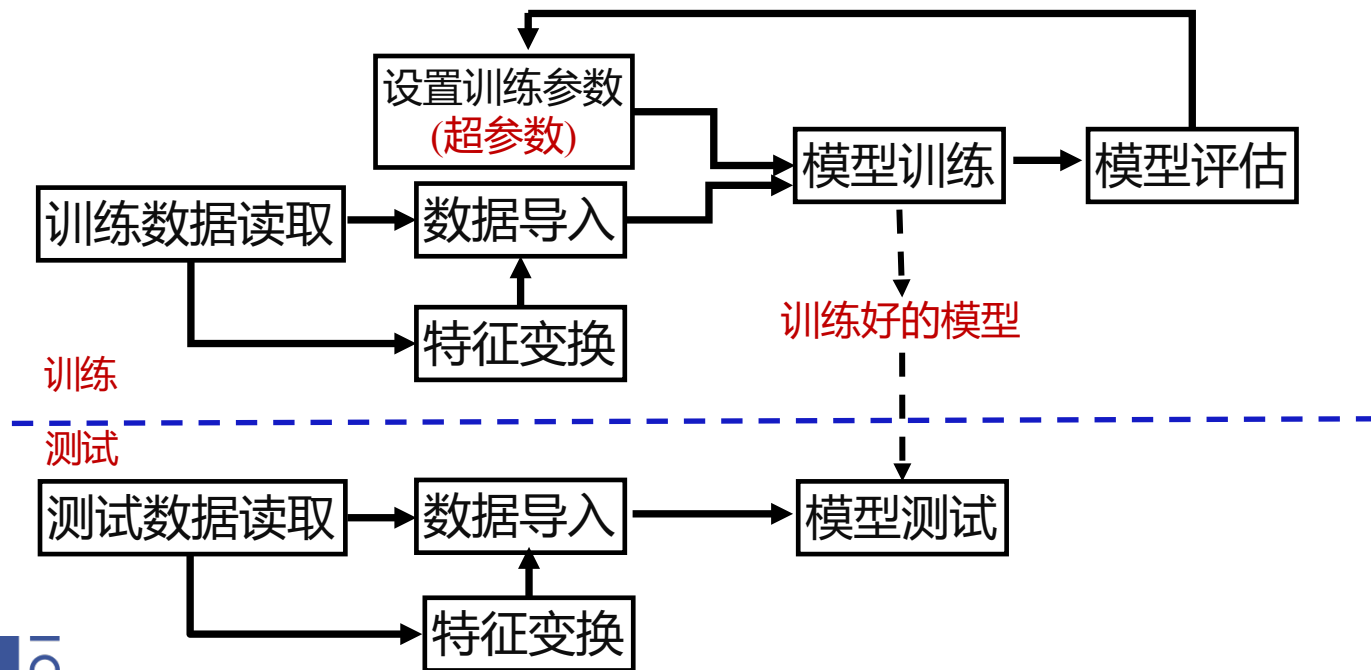
► Roadmap

- XGBoost简介
- XGBoost的优势
- 与XGBoost的第一次亲密接触
 - 直接调用XGBoost
 - 与scikit-learn一起使用



▶ 与XGBoost的第一次亲密接触

- 处理数据科学任务的一般流程



► 1. 确定任务

- 数据集：UCI机器学习库的Mushroom 数据集 (XGBoost安装包中的demo数据)
- 任务：根据蘑菇的22个特征判断蘑菇是否有毒
- 总样本数：8124
 - 可食用：4208 , 51.8%
 - 有毒：3916 , 48.2%
 - 训练样本：6513
 - 测试样本：1611
- 特征：Demo中22维特征经过处理，变成了126维特征量
 - xgboost/demo/data/featmap.txt 文件中有特征映射说明



<http://archive.ics.uci.edu/ml/machine-learning-databases/mushroom/>

<http://www.ai100.ai/>



► 2. 导入必要的工具包

- `import xgboost as xgb`
- `from sklearn.metrics import accuracy_score`
 - `sklearn(scikit-learn)`是一个流行的Python机器学习包

▶ 3.准备数据

- 将数据从文件中读出，并为XGBoost训练准备好
- # 设置数据路径，数据在xgbboost安装的路径下的demo目录
- # read in data，训练集和测试集
- my_workpath = './data/'
- dtrain = xgb.DMatrix(my_workpath + 'agaricus.txt.train')
- dtest = xgb.DMatrix(my_workpath + 'agaricus.txt.test')
- 该数据为libsvm格式的文本数据，libsvm的文件格式（稀疏特征）
 - 每一行为一个样本：1 101:1.2 102:0.03 0 1:2.1 10001:300 10002:400 ...
 - 开头的“1”是样本的标签。“101”和“102”为特征索引，'1.2'和'0.03'为特征的值。
 - 在两类分类中，用“1”表示正样本，用“0”表示负样本。也支持[0,1]表示概率用来做标签，表示为正样本的概率
- XGBoost加载的数据存储在对象Dmatrix中



► XGBoost的数据接口

- XGBoost加载的数据存储在对象Dmatrix中，做了存储效率和运行速度的优化
- 支持三种接口
 - libsvm txt格式文件
 - 常规矩阵（ numpy 2D array ）
 - xgboost binary buffer file

► 4. 设置训练参数

- *# specify parameters via map*
- `param = {'max_depth':2, 'eta':1, 'silent':0, 'objective':'binary:logistic' }`
- `max_depth` : 树的最大深度。缺省值为6，取值范围为： $[1, \infty]$
- `eta` : 为了防止过拟合，更新过程中用到的收缩步长。eta通过缩减特征的权重使提升计算过程更加保守。缺省值为0.3，取值范围为： $[0, 1]$
- `silent`: 0表示打印出运行时信息，取1时表示以缄默方式运行，不打印运行时信息。缺省值为0
- `objective` : 定义学习任务及相应的学习目标，“binary:logistic”表示二分类的逻辑回归问题，输出为概率。



► 5. 模型训练

- # 设置`boosting`迭代计算次数
- `num_round = 2`
- `bst = xgb.train(param, dtrain, num_round)`

► 6. 预测（训练数据上评估）

- 模型训练好后，可以用训练好的模型对进行预测
 - XGBoost预测的输出是概率，输出值是样本为第一类的概率 → 将概率值转换为0或1
- *# make prediction*
- `train_preds = bst.predict(dtrain)`
- `train_predictions = [round(value) for value in train_preds]`
- `y_train = dtrain.get_label()`
- `train_accuracy = accuracy_score(y_train, train_predictions)`
- `print ("Train Accuary: %.2f%%" % (train_accuracy * 100.0))`



► 6. 预测

- 模型训练好后，可以用训练好的模型对测试数据进行预测
- *# make prediction*
- `preds = bst.predict(dtest)`
- `predictions = [round(value) for value in preds]`
- `y_test = dtest.get_label()`
- `test_accuracy = accuracy_score(y_test, predictions)`
- `print("Test Accuracy: %.2f%%" % (test_accuracy * 100.0))`

▶ 示例代码

- 1_mushroom_baseline.ipynb

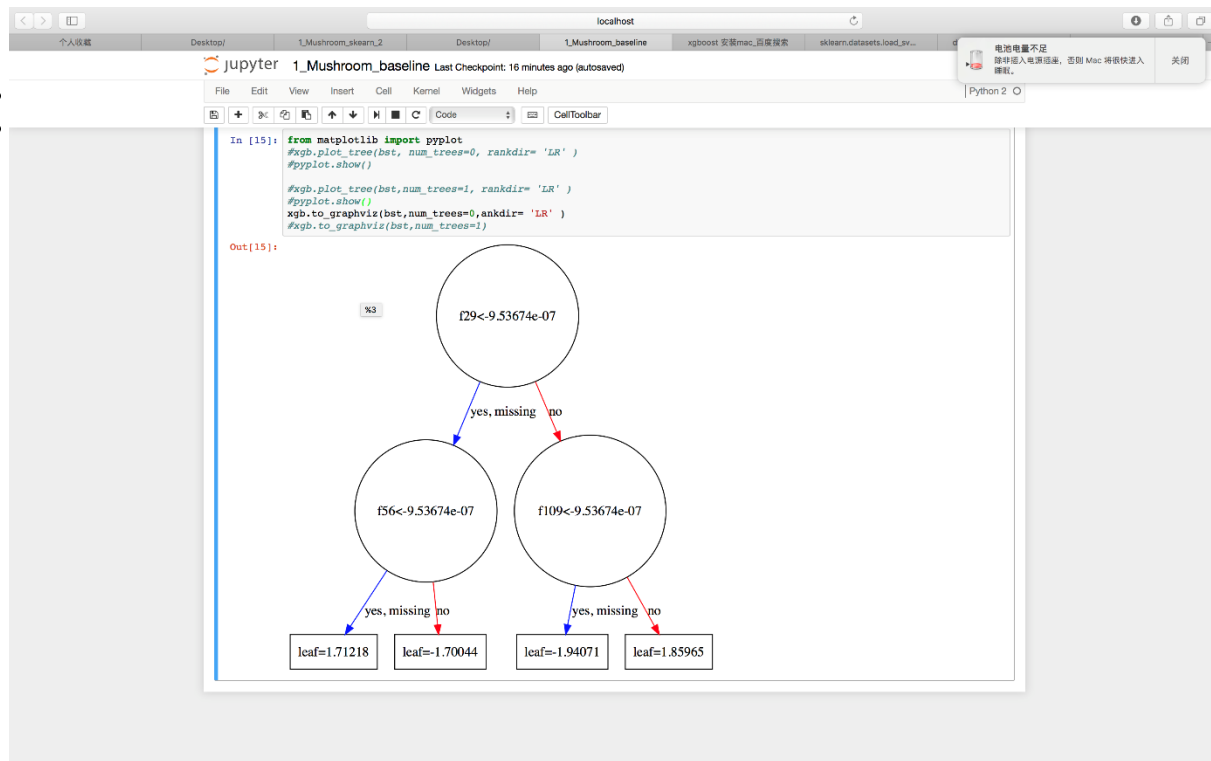
▶ 番外：模型可视化

- 可视化模型中的单颗树：调用XGBoost 的API `plot_tree()` / `to_graphviz()`
 -
- `from xgboost import plot_tree`
- `xgb.plot_tree(bst, num_trees=0, rankdir= 'LR')`
 - 第一个参数为训练好的模型
 - 第二个参数为要打印的树的索引（从0开始）
 - 第三个参数是打印的格式



▶ 树结构图

- 第一棵树：

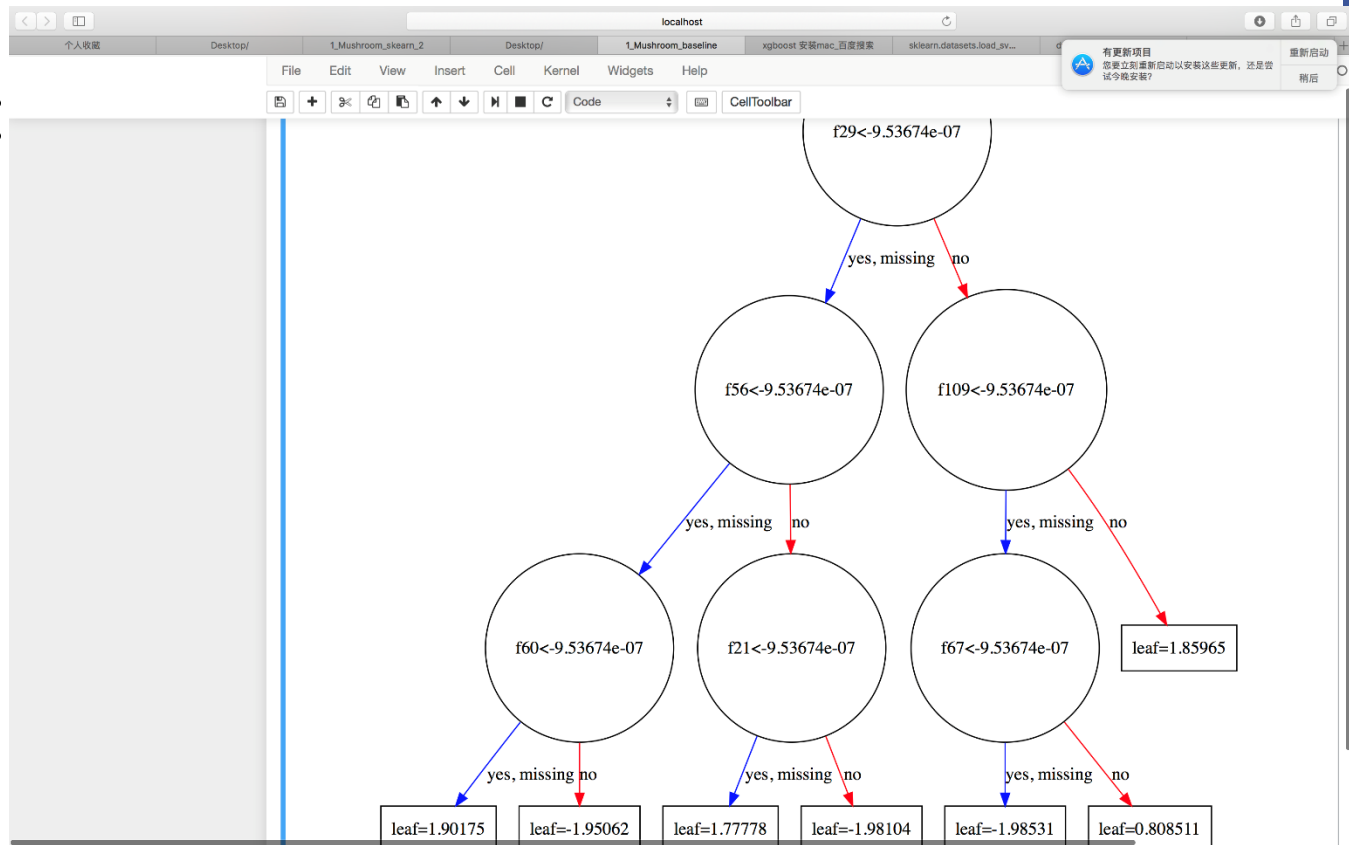


▶ 训练参数稍加变化...

- $\text{max_depth} = 2, \text{num_round} = 2$
 - 训练集上正确率 : 97.77%
 - 测试集上正确率 : 97.83%
- $\text{max_depth} = 3, \text{num_round} = 2$
 - 训练集上正确率 : 99.88%
 - 测试集上正确率 : 100.00 %

▶ 树结构图

- 第一棵树：



► 问题

- XGBoost有多少参数？这些参数怎么确定？
- 训练好的模型到底好不好？
 - 在测试数据上检验
- 在实际问题和Kaggle等竞赛中，只发布了训练数据，测试数据不可见
 - 训练-校验
- 我已经对其他机器学习算法很熟悉，scikit-learn框架使用熟练，怎样快速集成新XGBoost工具？



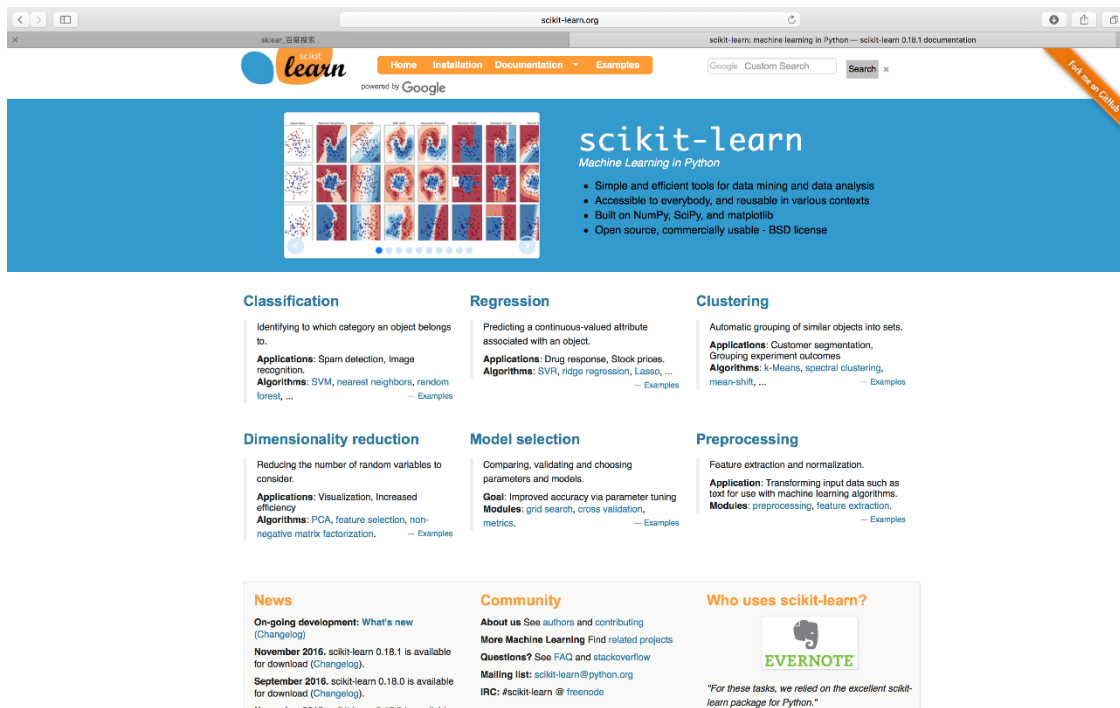
► Roadmap

- XGBoost简介
- XGBoost的优势
- 与XGBoost的第一次亲密接触
 - 直接调用XGBoost
 - 与scikit-learn一起使用



► scikit-learn

—*Machine Learning in Python*



The screenshot shows the scikit-learn website in a web browser. The browser's address bar displays "scikit-learn.org". The website's navigation bar includes links for "Home", "Installation", "Documentation", and "Examples", along with a search bar. The main banner features the scikit-learn logo, a grid of 16 small plots showing various machine learning results, and the text "scikit-learn Machine Learning in Python". Below the banner, the website is organized into a grid of sections, each describing a different machine learning task and providing examples of applications and algorithms. At the bottom, there are sections for "News", "Community", and "Who uses scikit-learn?".

scikit-learn
Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification
Identifying to which category an object belongs to.
Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, ...

Regression
Predicting a continuous-valued attribute associated with an object.
Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, ...

Clustering
Automatic grouping of similar objects into sets.
Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, ...


Dimensionality reduction
Reducing the number of random variables to consider.
Applications: Visualization, Increased efficiency
Algorithms: PCA, feature selection, non-negative matrix factorization, ...

Model selection
Comparing, validating and choosing parameters and models.
Goal: Improved accuracy via parameter tuning
Modules: grid search, cross validation, metrics, ...

Preprocessing
Feature extraction and normalization.
Application: Transforming input data such as text for use with machine learning algorithms.
Modules: preprocessing, feature extraction, ...

News
On-going development: [What's new \(Changelog\)](#)
November 2016. scikit-learn 0.18.1 is available for download ([Changelog](#)).
September 2016. scikit-learn 0.18.0 is available for download ([Changelog](#)).
November 2015. scikit-learn 0.17.0 is available.

Community
About us See [authors](#) and [contributing](#)
More Machine Learning Find [related projects](#)
Questions? See [FAQ](#) and [stackoverflow](#)
Mailing list: scikit-learn@python.org
IRC: #scikit-learn @ [freenode](#)

Who uses scikit-learn?

"For these tasks, we relied on the excellent scikit-learn package for Python."

► 与XGBoost的第一次亲密接触（cont.）

- 还是以蘑菇数据为例，学习XGBoost与sklearn结合，重点体会
 - scikit-learn框架下机器学习模型的训练和测试流程
 - 构造学习器（设置参数） — fit — predict

▶ 与scikit-learn结合

- XGBoost提供一个wrapper类，允许模型可以和scikit-learn框架中其他分类器或回归器一样对待
- XGBoost中分类模型为XGBClassifier
 - 模型参数在构造时传递
 - `bst = XGBClassifier(max_depth=2, learning_rate=1, n_estimators=num_round, silent=True, objective='binary:logistic')`



► 模型训练和测试

- 模型训练

- `bst.fit(X_train, y_train)`

- 模型测试

- `y_pred = bst.predict(X_test)`
- `predictions = [round(value) for value in y_pred]`
- `test_accuracy = accuracy_score(y_test, predictions)`
- `print("Test Accuracy: %.2f%%" % (test_accuracy * 100.0))`



▶ 示例代码

- 1_mushroom_sklearn_baseline.ipynb

► 校验集

- 我们在训练集和测试集上都检查了模型的性能
 - 实际场景中测试数据是未知的，如何评估模型？
 - 答案：校验集
 - 校验集：将训练数据的一部分留出来，不参与模型参数训练。留出来的这部分数据称为**校验集**
 - 余下的数据训练模型，训练好的模型在校验集上测试
 - 校验集上的性能可视为模型在未知数据上性能的估计 → 选择在校验集上表现最好的模型
-
- from [sklearn.model_selection](#) import [train_test_split](#)
 - # split data into train and test sets, 1/3的训练数据作为校验数据
 - seed = 7
 - test_size = 0.33
 - X_train_part, X_validation, y_train_part, y_validation = [train_test_split](#) (X_train, y_train, test_size=test_size, random_state=seed)



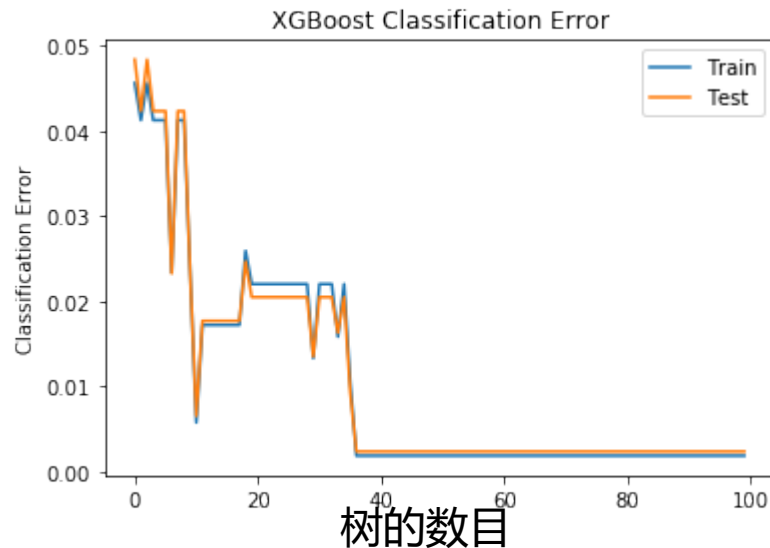
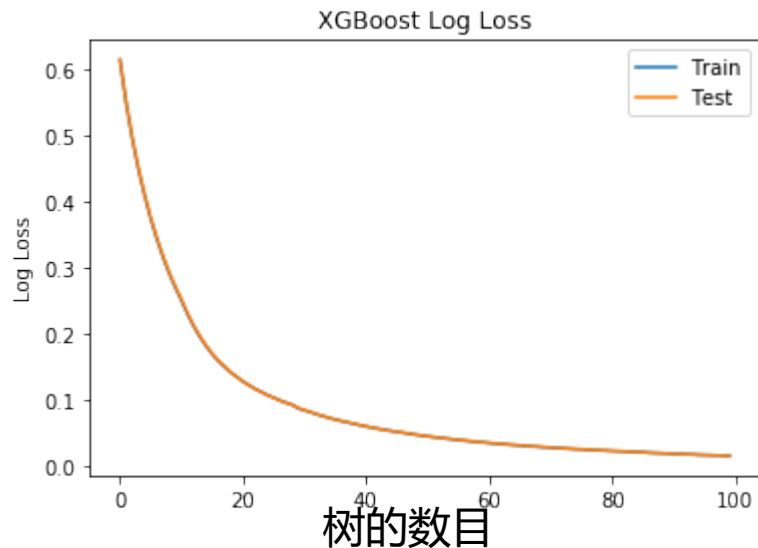
▶ 示例代码

- `1_mushroom_sklearn_split_baseline.ipynb`

► 学习曲线

- 学习曲线：模型预测性能随某个变化的学习参数（如训练样本数目、迭代次数）变化的情况
 - 例：XGBoost的迭代次数（树的数目）
- # 设置boosting迭代计算次数
- num_round = 100
- eval_set = [(X_train_part, y_train_part), (X_validation, y_validation)]
- bst.fit(X_train_part, y_train_part, eval_metric=["error", "logloss"], eval_set=eval_set, verbose=False)

▶ 学习曲线



由于Train和Test误差差异太小，
图形分辨率低，所以二者重合了



▶ 示例代码

- `1_mushroom_sklearn_split_learningcurve.ipynb`

► Early stop

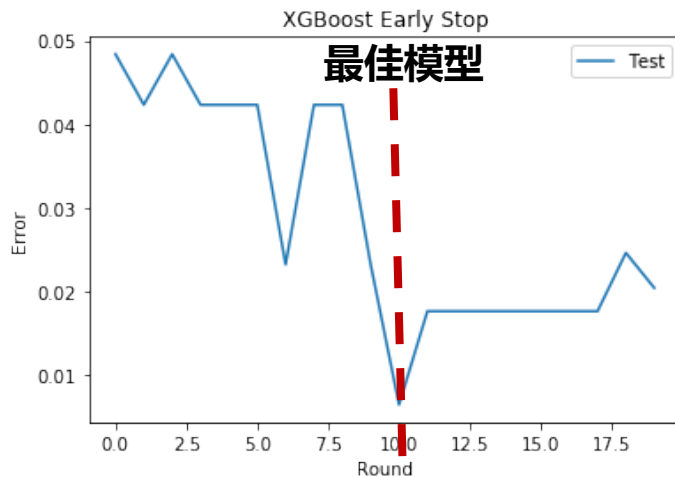
- Early stop: 一种防止训练复杂模型过拟合的方法
 - 监控模型在校验集上的性能：如果在经过固定次数的迭代，校验集上的性能不再提高时，结束训练过程
 - 当在测试集上的训练下降而在训练集上的性能还提高时，发生了过拟合
- # 设置boosting迭代计算次数
- `num_round = 100`
- `eval_set = [(X_validate, y_validate)]`
- `bst.fit(X_train, y_train, early_stopping_rounds=10, eval_metric="error", eval_set=eval_set, verbose=True)`



► Early stop (cont.)

- 例：

- [6] validation_0-error:0.023256
- [7] validation_0-error:0.042326
- [8] validation_0-error:0.042326
- [9] validation_0-error:0.023256
- [10] validation_0-error:0.006512
- [11] validation_0-error:0.017674
- [12] validation_0-error:0.017674
- [13] validation_0-error:0.017674
- [14] validation_0-error:0.017674
- [15] validation_0-error:0.017674
- [16] validation_0-error:0.017674
- [17] validation_0-error:0.017674
- [18] validation_0-error:0.024651
- [19] validation_0-error:0.020465
- [20] validation_0-error:0.020465
- Stopping. Best iteration:
- [10] validation_0-error:0.006512



▶ 示例源代码

- 1_mushroom_sklearn_split_earlystop.ipynb

交叉验证

- `train_test_split`将训练数据的一部分留出来做校验，不参与模型参数训练
 - 优点：速度快
 - 缺点：训练数据减少，一次校验集的划分会带来随机性
 - 答案：交叉验证 ([corss-valisation](#), CV)，但训练时间延长
 - 适合训练数据规模较大的情况（如上百万条记录）
 - 适合训练慢的机器学习模型
- **`k-折交叉验证`**：将训练数据等分成 k 份（ k 通常的取值为3、5或10）
 - 重复 k 次
 - 每次留出一份做校验，其余 $k-1$ 份做训练
 - k 次校验集上的平均性能视为模型在测试集上性能的估计
 - 该估计比`train_test_split`得到的估计方差更小

► 交叉验证(cont.)

- k -折交叉验证

- 重复 k 次

- 每次留出一份做校验，其余 $k-1$ 份做训练

- k 次校验集上的平均性能视为模型在测试集上性能的估计

- k 次结果可得到性能估计的均值和该估计的方差

- `from sklearn.model_selection import KFold`

- `from sklearn.model_selection import cross_val_score` #对给定参数的单个模型评估

- `kfold = KFold(n_splits=10, random_state=7)`

- `results = cross_val_score(model, X, Y, cv=kfold)`

- `print("Accuracy: %.2f%% (%.2f%%)" % (results.mean()*100, results.std()*100))`

- 注意：如果每类样本不均衡或类别数较多，采用`StratifiedKFold`，将数据集中每一类样本的数据等分



▶ 示例源代码

- 1_mushroom_sklearn_cv.ipynb

► GridSearchCV

- 参数调优GridSearchCV：我们可以根据交叉验证评估的结果，选择最佳参数的模型
 - 输入待调节参数的范围（grid），对一组参数对应的模型进行评估，并给出最佳模型及其参数
- `from sklearn.grid_search import GridSearchCV`
- `# 设置boosting迭代计算次数搜索范围`
- `param_test = { 'n_estimators': range(1, 51, 1)}`
- `clf = GridSearchCV(estimator = bst, param_grid = param_test, cv=5)`
- `clf.fit(X_train, y_train)`
- `clf.grid_scores_, clf.best_params_, clf.best_score_`



▶ 示例源代码

- 1_mushroom_sklearn_gridsearchcv.ipynb
- `GridSearchCV(estimator, param_grid, scoring=None, fit_params=None, n_jobs=1, iid=True, refit=True, cv=None, verbose=0, pre_dispatch='2*n_jobs', error_score='raise', return_train_score=True)`
 - Refit the best estimator with the entire dataset

► 模型评估小结

- 通常 k -折交叉验证是评估机器学习模型的黄金准则（ $k=3, 5, 10$ ）
- 当类别数目较多，或者每类样本数目不均衡时，采用stratified交叉验证
- 当训练数据集很大，train/test split带来的模型性能估计偏差很小，或者模型训练很慢时，采用train/test split 此时为hold-out
- 对给定问题找到一种技术，速度快且能得到合理的性能估计
- 如果有疑问，对回归问题，采用10-fold cross-validation；对分类，采用stratified 10-fold cross-validation

▶ 第一节总结

- What is XGBoost: GBM的优化实现
- Why XGBoost : 又快又好
- How to use XGBoost
 - 数据接口
 - 训练参数设置
 - 模型评估 (split, CV, metrics)
 - 与scikit-learn结合使用

▶ 下节课预告

- 监督学习
 - 模型
 - 参数
 - 目标函数
 - 损失函数
 - 正则
 - 优化
- 分类回归树(CART)
 - 模型
 - 参数
 - 目标函数
 - 优化
 - 分裂
 - 剪枝



THANK YOU

北京智百科技有限公司



100 AI100



<http://www.ai100.ai/>