CTR 广告点击率预估

背景

在线 OnLine 广告分类

- 展示类广告：腾讯的广点通---如 QQ 空间<参考提供资料>
- 搜索广告：百度蜂巢
- 社交平台上广告(Facebook,微博上广告)

常用的计费方式：

1. CPM(cost per mile) 按照展示收费，不管用户看到广告没，只要广告每天达到一定的曝光次数，就需要给钱，广告组会给平台组出钱
2. CPC(Cost per Click) 搜索广告中使用，按照点击收费----百度、Google 的收费方式
3. CPA(Cost Per Action) 必须用户点击了广告并下单之后才会收费的方式

$$CTR = \frac{Clicks}{Impression} \times 100\%$$

曝光了多少次的基础上有多少次的点击，1000 次的曝光下有 100 次的点击，CTR=0.1，在 CPC 计价模式下，每天点击广告的次数是有限的，一般点击率一天能达到 1%，CTR 越高越高，越高的商品优先得到推荐，但是也要结合 Price 价格来分析，这里对比两个场景**(1)CTR=0.5，Price=5 和(2)CTR=0.1,Price=1000**，我们优先选择(2)场景进行推荐，虽然点击率低。Price 一般是广告组和平台，以及有一些计价方式来确定价格，因此我们的业务场景仅需要确定 CTR 就可以。

在搜索或社交平台上，我们希望用户会点还是不会点，因此通过分析可以得到这是一个二分类问题。这里会用到那个模型呢，这里我们首先考虑那个算法会在具备数据集基础上能够输出一个概率值，这个算法一般使用 LR 逻辑斯特回归，还有基于树的模型以及 DNN 等，使用最多的是 LR，如腾讯广点通、百度蜂巢。

LR 有模型简单，训练快，可解释性非常高的优点。在一般大公司会上 DNN，但是不会立马取掉 LR，因为 LR 可解析性很高，同时小公司一般很少使用。

备注：天池广告比赛

https://tianchi.aliyun.com/competition/information.htm?spm=5176.100067.5678.2.29284558Z5rEhk&raceId=231647

1 基本介绍

CTR（Click-Through-Rate）互联网广告常用的术语，指网络广告（图片广告/文字广告/关键词广告/排名广告/视频广告等）的点击到达率，即该广告的点击量（严格的来说，可以是到达目标页面的数量）除以广告的浏览量（PV- Page View）。

CTR=点击量/浏览量

CTR 是衡量互联网广告效果的一项重要指标。

CTR 指在搜索引擎中输入关键词后进行搜索，然后按竞价等因素把相关的网页按顺序进行排列出来，然后用户会选择自己感兴趣的网站点击进去；把一个网站所有搜索出来的次数作为总次数，把用户点击并进入网站的次数占总次数的比例叫点击率。

较低的点击率意味着，不管您的网站排名如何靠前，用户都不会点击它。这可能说明，他们不认为您的网站会满足他们的需求，或其他网站看起来更好些。

2 计算公式

计算公式为 CTR=点击量/展示量，即 Click / Show content。

CTR：点击率，Click-Through-Rate (点击通过比率)

3 基本用法

随着投放数据的进行，可以点击率（CTR）以及平均点击价格（CPC）为主要效果评判指标，筛选出效果不好的关键词，分析原因并给出提高 CTR 降低 CPC 的方案。下面是针对一些常见情况给出的建议。

点击率过小时，是由两种原因造成的，展现量过小，或点击数偏低。

1、展现量低，进而点击数也小。展现量过低说明潜在受众搜索需求发生的较少，也即推广结果展现在潜在受众前的机会较少，推广商户可以通过拓展关键词来提高展现量，即提高推广信息展现的机会。

2、展现量高，但是点击数偏低，造成点击率（CTR）偏低。

点击率（CTR）的偏低说明可能关键词与文案的相关性不高，所以无法满足潜在受众的需求，进而点击数小。可以通过改善文案写作，提高关键词与文案的相关性来提高点击率（CTR）。

点击率（CTR）的偏低也说明可能推广结果的平均排名较低，不具有竞争力。可以通过调高平均点击价格（CPC）来提高排名。

点击率（CTR）的偏低还可能说明关键词匹配模式的问题。例如，推广商户购买了“葡萄”等相关关键词，用户在搜索“葡萄牙”时商户的推广结果也可能会出现，这时推广结果就是无效展现，即为推广结果信息没有展现在潜在受众前。此种情况就需通过“否定匹配”模式来解决，将“葡萄牙”设置为否定匹配，即为用户在搜索“葡萄牙”时，推广商户的推广结果不会展现，降低了无效展现的风险。

Avazu 实战

概述

在线广告中，点击率（CTR）是评估广告效果的重要指标。因此，点击预测系统是必不可少的，并广泛用于赞助搜索和实时出价。为了这次比赛，我们提供了 **11 天的 Avazu 数据来构建和测试预测模型**。你能找到一个战胜标准分类算法的策略吗？本次比赛的获奖模型将在开源许可下发布。

提交格式如下：**id 商品+点击率预估值**

Submissions are evaluated using the Logarithmic Loss (smaller is better).

Submission Format

The submissions should contain the predicted probability of click for each ad impression in the test set using the following format:

```
id,click
60000000,0.384
63895816,0.5919
759281658,0.1934
895936184,0.9572
...
```

File descriptions

train - Training set. 10 days of click-through data, ordered chronologically. Non-clicks and clicks are subsampled according to different strategies.

训练集。10 天的点击数据，按时间顺序排列。根据不同的策略对非点击和点击进行二次抽样。

test - Test set. 1 day of ads to for testing your model predictions.

测试集。1 天的广告来测试您的模型预测。

sampleSubmission.csv - Sample submission file in the correct format, corresponds to the All-0.5 Benchmark.

Data fields

id: ad identifier 广告标识符

click: 0/1 for non-click/click 0/1 进行非点击/点击

hour: format is YYMMDDHH, so 14091123 means 23:00 on Sept. 11, 2014 UTC.格式为 YYMMDDHH，因此 14091123 表示 2014 年 9 月 11 日 23:00 时的 UTC。

C1 -- anonymized categorical variable 匿名分类变量
banner_pos
site_id
site_domain
site_category
app_id
app_domain
app_category
device_id
device_ip
device_model
device_type
device_conn_type
C14-C21 -- anonymized categorical variables 匿名分类变量

数据 EDA 分析

CTR Prediction

https://www.kaggle.com/c/avazu-ctr-prediction/data

File descriptions

train - Training set. 10 days of click-through data, ordered chronologically. Non-clicks and clicks are subsampled according to different strategies.

https://www.kaggle.com/c/avazu-ctr-prediction/download/train.gz

test - Test set. 1 day of ads to for testing your model predictions.

https://www.kaggle.com/c/avazu-ctr-prediction/download/test.gz

sampleSubmission.csv - Sample submission file in the correct format, corresponds to the All-0.5 Benchmark.

https://www.kaggle.com/c/avazu-ctr-prediction/download/sampleSubmission.gz

Data fields

id: ad identifier click: 0/1 for non-click/click hour: format is YYMMDDHH, so 14091123 means 23:00 on Sept. 11, 2014 UTC.

C1 -- anonymized categorical variable, banner_pos, site_id, site_domain, site_category, app_id, app_domain, app_category, device_id, device_ip, device_model, device_type, device_conn_type, C14-C21 -- anonymized categorical variables

Load Data

In [1]:
```
import pandas as pd


# Initial setup
train_filename = "train_small.csv"
test_filename = "test.csv"
submission_filename = "submit.csv"

training_set = pd.read_csv(train_filename)
```

```
/Library/Python/2.7/site-packages/IPython/core/interactiveshell.py:2723: Dt
import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

Explore Data

In [3]:
```
training_set.head(10)
```

Out[3]:

	id	click	hour	C1	banner_pos	site_id	site_domain	site_categor
0	1000009418151094273	0	14102100	1005	0	1fbe01fe	f3845767	28905ebd
1	10000169349117863715	0	14102100	1005	0	1fbe01fe	f3845767	28905ebd
2	10000371904215119486	0	14102100	1005	0	1fbe01fe	f3845767	28905ebd
3	10000640724480838376	0	14102100	1005	0	1fbe01fe	f3845767	28905ebd
4	10000679056417042096	0	14102100	1005	1	fe8cc448	9166c161	0569f928
5	10000720757801103869	0	14102100	1005	0	d6137915	bb1ef334	f028772b
6	10000724729988544911	0	14102100	1005	0	8fda644b	25d4cfcd	f028772b
7	10000918755742328737	0	14102100	1005	1	e151e245	7e091613	f028772b
8	10000949271186029916	1	14102100	1005	0	1fbe01fe	f3845767	28905ebd
9	10001264480619467364	0	14102100	1002	0	84c7ba46	c4e18dd6	50e219e0

10 rows × 24 columns

In [3]:
```
training_set.describe()
```

Out[3]:

	click	hour	C1	banner_pos	device_type	device_conn_type	C
count	99999.000000	99999.0	99999.000000	99999.000000	99999.000000	99999.000000	9
mean	0.174902	14102100.0	1005.034440	0.198302	1.055741	0.199272	1
std	0.379885	0.0	1.088705	0.402641	0.583986	0.635271	3
min	0.000000	14102100.0	1001.000000	0.000000	0.000000	0.000000	3
25%	0.000000	14102100.0	1005.000000	0.000000	1.000000	0.000000	1
50%	0.000000	14102100.0	1005.000000	0.000000	1.000000	0.000000	1
75%	0.000000	14102100.0	1005.000000	0.000000	1.000000	0.000000	2
max	1.000000	14102100.0	1010.000000	5.000000	5.000000	5.000000	2

建模实战

In [2]:
```
# id: ad identifier
# click: 0/1 for non-click/click
# hour: format is YYMMDDHH, so 14091123 means 23:00 on Sept. 11, 2014 UTC.
# C1 -- anonymized categorical variable
# banner_pos
# site_id
# site_domain
# site_category
# app_id
# app_domain
# app_category
# device_id
# device_ip
# device_model
# device_type
# device_conn_type
# C14-C21 -- anonymized categorical variables
from sklearn.externals import joblib
from sklearn.cross_validation import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import metrics

from utils import load_df
```

In [3]:
```
# 结果衡量
def print_metrics(true_values, predicted_values):
    print "Accuracy: ", metrics.accuracy_score(true_values, predicted_values)
    print "AUC: ", metrics.roc_auc_score(true_values, predicted_values)
    print "Confusion Matrix: ", + metrics.confusion_matrix(true_values, predicted_values)
    print metrics.classification_report(true_values, predicted_values)

# 拟合分类器
def classify(classifier_class, train_input, train_targets):
    classifier_object = classifier_class()
    classifier_object.fit(train_input, train_targets)
    return classifier_object

# 模型存储
def save_model(clf):
    joblib.dump(clf, 'classifier.pkl')
```

In [4]:
```
train_data = load_df('train_small.csv').values
```

```
D:\ProgramCJ\Anaconda\Anaconda2\lib\site-packages\IPython\core\interactiveshell.py:2825:
dtype option on import or set low_memory=False.
  if self.run_code(code, result):
```

In [5]:
```
train_data[:,:]
```

Out[5]:
```
array([[       0, 14102100,     1005, ...,       35,       -1,       79],
       [       0, 14102100,     1005, ...,       35,   100084,       79],
       [       0, 14102100,     1005, ...,       35,   100084,       79],
       ...,
       [       0, 14102100,     1005, ...,       35,       -1,       79],
       [       1, 14102100,     1005, ...,       35,       -1,       79],
       [       0, 14102100,     1005, ...,       35,       -1,       79]], dtype=int64)
```

In [6]:
```
# 训练和存储模型
X_train, X_test, y_train, y_test = train_test_split(train_data[0::, 1::], train_data[0::, 0],
                                                    test_size=0.3, random_state=0)

classifier = classify(LogisticRegression, X_train, y_train)
predictions = classifier.predict(X_test)
print_metrics(y_test, predictions)
save_model(classifier)
```

```
Accuracy:  0.8233
AUC:  0.5
Confusion Matrix:  [[24699     0]
 [ 5301     0]]
             precision    recall  f1-score   support

          0       0.82      1.00      0.90     24699
          1       0.00      0.00      0.00      5301

avg / total       0.68      0.82      0.74     30000
```

In [8]:
```
# 按照指定的格式生成结果
def create_submission(ids, predictions, filename='submission.csv'):
    submissions = np.concatenate((ids.reshape(len(ids), 1), predictions.reshape(len(predictions), 1)), axis=1)
    df = DataFrame(submissions)
    df.to_csv(filename, header=['id', 'click'], index=False)
```

In [11]:
```
import numpy as np
from pandas import DataFrame

classifier = joblib.load('classifier.pkl')
test_data_df = load_df('test.csv', training=False)
ids = test_data_df.values[0:, 0]
predictions = classifier.predict(test_data_df.values[0:, 1:])
create_submission(ids, predictions)
```

代码详解：

```
    #1.导入数据并进行简单的数据探索
import os
data_path = os.path.join(".", "train_small.csv")
import pandas as pd
ctr_data1 = pd.read_csv(data_path)
print(ctr_data1.shape)
# print ctr_data.head()
# print ctr_data.describe()
print (ctr_data1.columns)
print ("="*100)
training_Set=ctr_data1.drop(['id','site_id', 'app_id', 'device_id', 'device_ip', 'site_domain',
                             'site_category', 'app_domain', 'app_category', 'device_model'], axis=1)
ctr_data=training_Set.values
#2.对数据进行处理和分析
from sklearn.model_selection import train_test_split
X=ctr_data[:,1:]
print (X.shape)
y=ctr_data[:,0]
print (y.shape)
X_train, X_test, y_train, y_test=train_test_split(X,y,test_size=0.22,random_state=33)
print (X_train.shape)
```

```
print (y_train.shape)
# #3.引入机器学习算法
from sklearn.linear_model import LogisticRegression
# lr=LogisticRegression()
#                  0          0.83          1.00          0.91        18240
#                  1          0.00          0.00          0.00         3760
#
# avg / total            0.69          0.83          0.75        22000
lr=LogisticRegression(C=0.1, penalty= 'l1')    #预测效果提示到 77%
lr.fit(X_train,y_train)
# #4.模型预测
y_pred=lr.predict(X_test)
print (y_pred)
# # #5.模型校验
print( lr.score(X_train,y_train))
print (lr.score(X_test,y_test))
from sklearn.metrics import confusion_matrix
print( confusion_matrix(y_test,y_pred))
from sklearn.metrics import classification_report
print( classification_report(y_test,y_pred))
# #6.保存模型
from sklearn.externals import joblib
joblib.dump(lr,filename="Ctr_Predict.pkl")
# #8.按照要求写入对应的 csv 文件
import numpy as np
import pandas as pd
ctr_data2=pd.read_csv("test.csv")
ctr_data3=ctr_data2.drop(['click','site_id', 'app_id', 'device_id', 'device_ip', 'site_domain',
                          'site_category', 'app_domain', 'app_category', 'device_model'], axis=1)
print( ctr_data3)
ids=ctr_data3.values[0:,0]
y_pred_test=lr.predict(ctr_data3.values[0:,1:])
# # # print ids
submit=np.concatenate((ids.reshape(len(ids),1),y_pred_test.reshape(len(y_pred_test),1)),axis=1)
df=pd.DataFrame(submit)
df.to_csv("submit.csv", header=['id', 'click'], index=False)
```

展现量、点击量、点击率；访客数、访问次数、浏览量的区别与作用

1.什么是展现量、点击量、点击率

在百度推广后台可以看到消费、平均价格、点击、展现、点击率、千次展现费用等数据，这些数据是你全面评估推广效果、深入开展推广优化的基础。

在网民搜索查询时，如果您账户内符合网民搜索需求的关键词被触发，该关键词所对应的创意将出现在搜索结果页，称之为关键词和创意的一次展现。一段时间内您获得的展现次数称之为 “展现量”。展现量体现了你的关键词质量度和创意的好坏（如果你的创意展现方式是优先展现的话）。

在您的推广结果展现时，如果网民对您的推广结果感兴趣，希望进一步地了解您的产品/服务，可能将会点击访问您的网站。一段时间内您获得的点击次数称之为“点击量”。简单的说，点击量指你的创意被点击的次数。

点击量/展现量=点击率。点击率体现你的创意的吸引力。

2.如何用好展现量、点击率让推广更高效？

展现量有助于您了解推广结果覆盖了多少网民，是一个数量上的概念。通过统计报告提供的展现量数据，您可以发现哪些关键词创意的展现机会较大，每天能够给您带来多少次的曝光机会，从而估算出您的推广活动能够覆盖到多少数量的潜在客户。

如果您的所有关键词每天合计的展现量一直比较少，无法让您的推广结果得到充分的曝光，那么我们建议您可以考虑提交更多的关键词，并采用广泛匹配，从而达到覆盖更多潜在客户的目的。

3.什么是访客数(UV)

访客数就是指一天之内到底有多少不同的用户访问了你的网站。访客数要比 IP 数更能真实准确地反映用户数量。百度统计完全抛弃了 IP 这个指标，而启用了访客数这一指标，是因为 IP 往往不能反映真实的用户数量。尤其对于一些流量较少的企业站来说，IP 数和访客数会有一定的差别。

访客数主要是以 cookie 为依据来进行判断的，而每台电脑的 cookie 也是不一样的。有些情况下 IP 数会大于真实的访客数。有时候访客数也会大于 IP 数。访客数要比 IP 数更能真实准确地反映用户数量。

4.什么是访问次数

访问次数是指访客完整打开了网站页面进行访问的次数。访问次数是网站的访问速度的衡量标准

如果访问次数明显少于访客数，就说明很多用户在没有完全打开网页时就将网页关闭了。如果是这样的情况，我们就要好好检查一下网站的访问速度了，看看到底是网站空间出了问题还是网站程序出了问题。访问次数一般会大于访客数。

5.什么是浏览量(PV)

浏览量和访问次数是呼应的。用户访问网站时每打开一个页面，就记为 1 个 PV。同一个页面被访问多次，浏览量也会累积。一个网站的浏览量越高，说明这个网站的知名度越高，内容越受用户喜欢。一味地重视 PV 也是没有太大意义的（PV 跟点击量差不多吧）。

PV 是一个重要的指标，反映了网站内容是否对用户有足够的吸引力。对于竞价而言，只能是侧面反映，因为我们设置了访问 URL。很多用户需求也非常明确，来到网站之后，往往只会寻找自己需求的产品，所以一味地重视 PV 也是没有太大意义的。应该把重点内容展示给目标客户就可以了，就没必要一味地追求 PV 值，追求那些转化率、跳出率、UV、转化次数等那才是重点。

6.什么是转化次数

（重要，但是对于竞价，一般是把点击商务通作为转化页面的，所以很多是无意点击）

潜在用户在我们的网站上完成一次我们期望的行为，就叫做一次转化。我们可以在百度统计的后台设置相应的转化页面，用户访问这个页面 1 次，就记为 1 次转化。

7.什么是转化率：

转化率=转化次数/访问次数。

对竞价而言，是关键词和访问页面的精准的指标。转化率可以用来衡量网络营销的效果。如果我们在 A、B 两个网站同时投放了广告，A 网站每天能带来 100 次用户访问，但是只有 1 个转化，B 网站每天能带来 10 次用户访问，但是却有 5 个转化。这就说明 B 网站带来的转化率更高，用户更加精准，网络营销效果更好。

8.什么是平均访问时长

平均访问时长是衡量网站用户体验的一个重要指标

平均访问时长是用户访问网站的平均停留时间。平均访问时长=总访问时长/访问次数。如果用户不喜欢网站的内容，可能稍微看一眼就关闭网页了，那么平均访问时长就很短;如果用户对网站的内容很感兴趣，一连看了很多内容，或者在网站停留了很长时间，平均访问时长就很长。

9.什么叫平均访问页数

平均访问页数也是衡量网站的用户体验的指标

平均访问页数是用户访问网站的平均浏览页数。平均访问页数=浏览量/访问次数。平均访问页数很少，说明访客进入你的网站后访问少数几个页面就离开了。

10.什么是跳出率

跳出率是反映网站流量质量的重要指标

跳出率是指访客来到网站后，只访问了一个页面就离开网站的访问次数占总访问次数的百分比。跳出率=只访问一个页面就离开网站的访问次数/总访问次数，跳出率越低说明流量质量越好，用户对网站的内容越感兴趣。

分析总结

1.需要分析创意的展现量、点击量、点击率，判断创意是否有吸引力；

2.需要分析关键词的转化次数、转化率，判断关键词是否精准；

3.需要分析访问次数和访客数，如果访问次数明显少于访客数，就说明很多用户在没有

完全打开网页时就将网页关闭了。如果是这样的情况，我们就要好好检查一下网站的访问速度了，看看到底是网站空间出了问题还是网站程序出了问题。

3.把平均访问页数和平均访问时长这两个指标放在一起来衡量网站的用户体验：如果平均访问页数较少，平均访问时长较短，就要分析以下几个问题：是否需要否关键词;网站的访问速度如何;用户进入网站后能否找到需要的内容（访问 URL 的设置）;网站内容对用户是否有吸引力。

4.网站跳出率和平均访问时长可以反映出网站推广关键词的选择是否精准，创意的撰写是否优秀，着陆页的设计是否符合用户体验。

Kaggler

https://github.com/jeongyoonlee/Kaggler

Kaggler - Kaggler 的 Python 包

分享一下我在网上学习的东西，作为一个名为 Kaggler 的 Python 软件包。

您可以使用 pip 安装它，如下所示：

```
$ pip install -U Kaggler
```

那么，导入算法类如下：

```
from kaggler.online_model import SGD, FTRL, FM, NN, NN_H2
```

目前它支持 4 种在线学习算法 - SGD，FTRL，FM（原始版本），NN（1 或 2 个 ReLU 隐藏层）。

它使用 liblinear 样式稀疏输入格式 - 选择相同的输入文件，以便可以在 XGBoost，VW （稍有修改）， libFM，SVMLight 等其他常用工具中使用相同的输入文件。

代码和示例可在 https://github.com/jeongyoonlee/Kaggler 获得，包文档可在 http://pythonhosted.org//Kaggler/获得。

我计划在我参加更多比赛并了解更多信息时对其进行更新 - 支持 CSV 输入格式并添加 Daniel Yoo 的 FM-FTRL 将是我的下一次更新。

Kaggler

Kaggler is a Python package for lightweight online machine learning algorithms and utility functions for ETL and data analysis. It is distributed under the version 3 of the GNU General Public License.

Its online learning algorithms are inspired by Kaggle user tinrtgu's code. It uses the sparse input format that handles large sparse data efficiently. Core code is optimized for speed by using Cython.

Algorithms

Currently algorithms available are as follows:

Online learning algorithms

- Stochastic Gradient Descent (SGD)
- Follow-the-Regularized-Leader (FTRL)
- Factorization Machine (FM)
- Neural Networks (NN) - with a single (NN) or two (NN_H2) ReLU hidden layers
- Decision Tree

Batch learning algorithm

- Neural Networks (NN) - with a single hidden layer and L-BFGS optimization

Dependencies

Python packages required are listed in `requirements.txt`

- cython
- h5py
- numpy/scipy
- pandas
- scikit-learn
- ml_metrics

Installation

Using pip

Python package is available at PyPi for pip installation:

```
sudo pip install -U Kaggler
```

If installation fails because it cannot find `MurmurHash3.h`, please add `.` to `LD_LIBRARY_PATH` as described here.

From source code

If you want to install it from source code:

```
python setup.py build_ext --inplace
sudo python setup.py install
```

Input Format

Kaggler supports CSV (`.csv`), LibSVM (`.sps`), and HDF5 (`.h5`) file formats:

```
# CSV format: target,feature1,feature2,...
1,1,0,0,1,0.5
0,0,1,0,0,5

# LibSVM format: target feature-index1:feature-value1 feature-index2:feature-value2
1 1:1 4:1 5:0.5
0 2:1 5:1

# HDF5
issparse: binary flag indicating whether it stores sparse data or not.
target: stores a target variable as a numpy.array
shape: available only if issparse == 1. shape of scipy.sparse.csr_matrix
indices: available only if issparse == 1. indices of scipy.sparse.csr_matrix
indptr: available only if issparse == 1. indptr of scipy.sparse.csr_matrix
data: dense feature matrix if issparse == 0 else data of scipy.sparse.csr_matrix
```

Example：

```
from kaggler.online_model import SGD, FTRL, FM, NN
# SGD
clf = SGD(a=.01,                    # learning rate
          l1=1e-6,                  # L1 regularization parameter
          l2=1e-6,                  # L2 regularization parameter
          n=2**20,                  # number of hashed features
          epoch=10,                 # number of epochs
```

```
                    interaction=True)         # use feature interaction or not
    # FTRL
    clf = FTRL(a=.1,                          # alpha in the per-coordinate rate
                    b=1,                      # beta in the per-coordinate rate
                    l1=1.,                    # L1 regularization parameter
                    l2=1.,                    # L2 regularization parameter
                    n=2**20,                  # number of hashed features
                    epoch=1,                  # number of epochs
                    interaction=True)         # use feature interaction or not
    # FM
    clf = FM(n=1e5,                           # number of features
                epoch=100,                    # number of epochs
                dim=4,                        # size of factors for interactions
                a=.01)                        # learning rate
    # NN
    clf = NN(n=1e5,                           # number of features
                epoch=10,                     # number of epochs
                h=16,                         # number of hidden units
                a=.1,                         # learning rate
                l2=1e-6)                      # L2 regularization parameter
    # online training and prediction directly with a libsvm filefor x, y in
clf.read_sparse('train.sparse'):
        p = clf.predict_one(x)          # predict for an input
        clf.update_one(x, p - y)        # update the model with the target using error
    for x, _ in clf.read_sparse('test.sparse'):
        p = clf.predict_one(x)
    # online training and prediction with a scipy sparse matrixfrom kaggler import load_data

    X, y = load_data('train.sps')

    clf.fit(X, y)
    p = clf.predict(X)
```