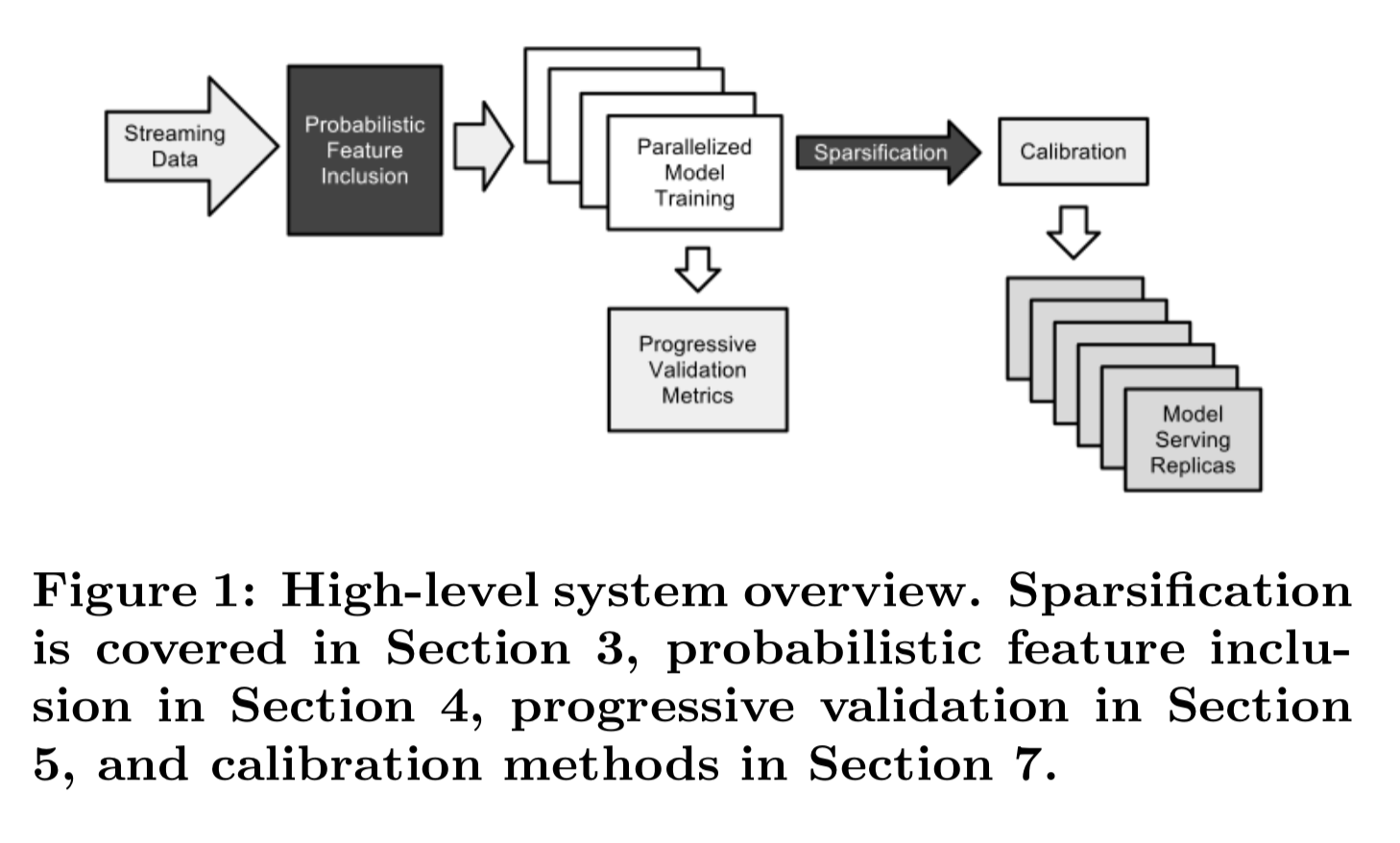
Title: Ad Click Prediction : a view from the trenches

Author: H.Brendan McMahan

Source:Google

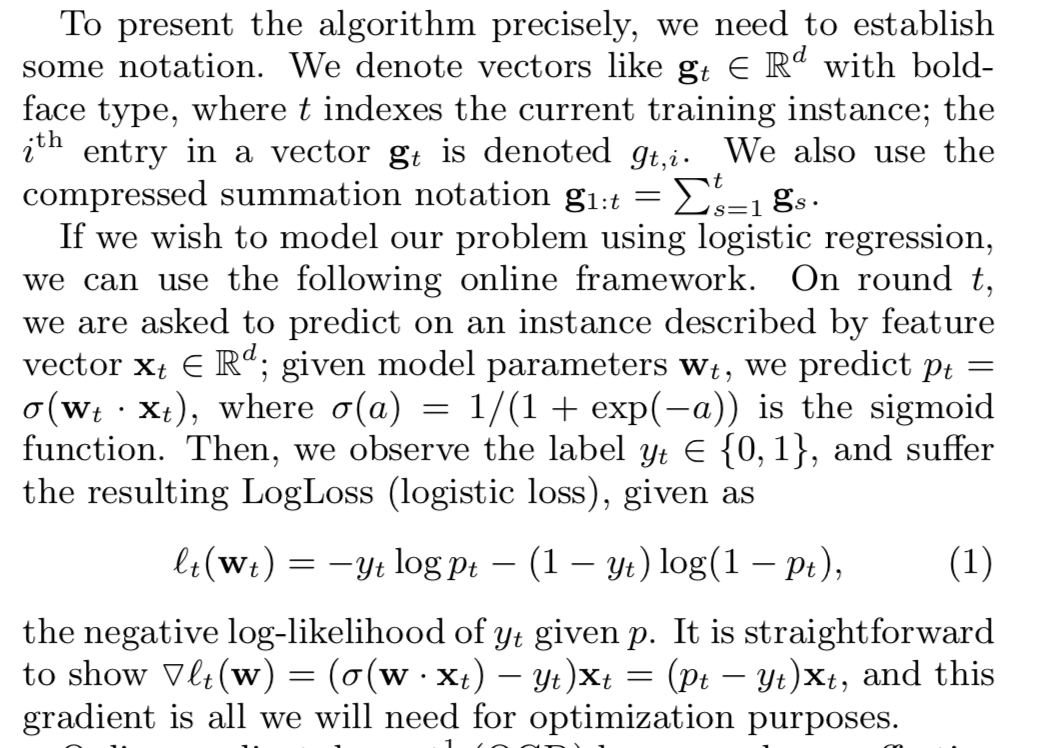
Abstract:

Introduction:

Brief system overview:  


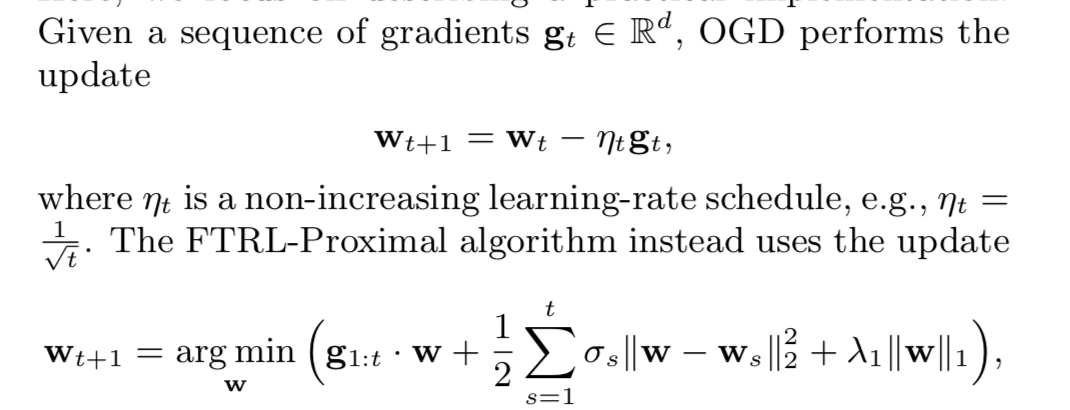
Sparsification : 进行稀疏化

Calibration: 进行矫正

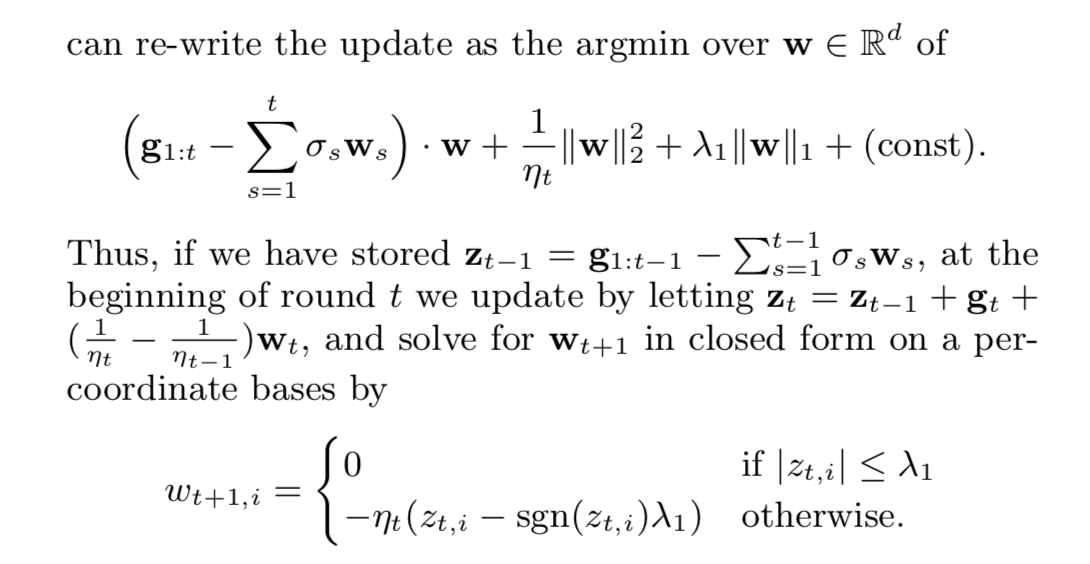
Online learning and sparsity:  


The number of the non-zero coefficients in w is the determining factor of memory usage.

FTRL算法不需要正则化的时候，相当于OGD（online gradient descent）算法, 但是因为FTRL算法使用了另外一种lazy representation of the model coficients w，L1 正则化可以被更高效的实现。



这两个形式看似不一样，实际上只要取lamta的值为0的话，以上两者是一样的。

其中的FTRL-proximal算法可以重写成以下形式：  


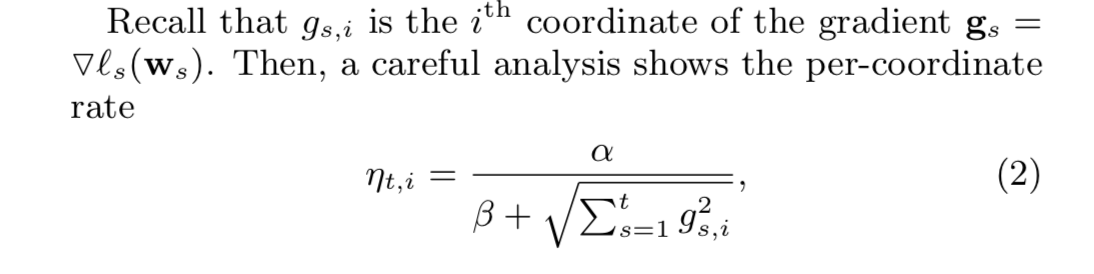
从以上的形式可以很容易发现当lamba1=0的时候：



这就是OGD的表达形式

当然FTRL除了可以加入L1正则项还可以加入L2正则项

Experimental results:  
straw-man 算法，simply maintain a count of the number of times it has been a feature. Until that count pass a threshold k, online gradient descent proceeds as usual. FTRL 算法improved sparsity with the same or better prediction accuracy.

Per-coordinate learning rates:  


这个算法要求我们记录both the sum of the gradients and the sum of the squares of the gradients for each feature. 为了节约空间，提出了一个memory-saving formulation的方式来节省计算the sum of the squares of the gradients

Saving memory at massive scale

4.1 probabilistic feature inclusion

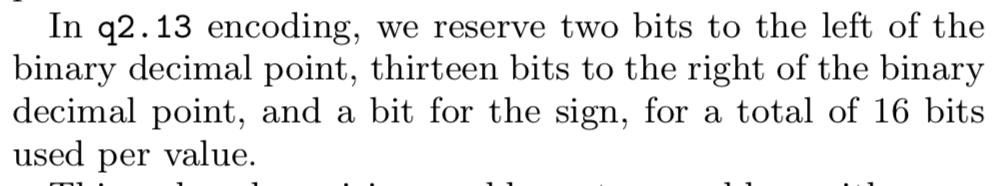
Probabilistic feature inclusion, in which new features are included in the model probabilistically as they first occur. This achieves the effect of pre-processing the data, but can be executed in an online setting.

这种方式有两种方法：

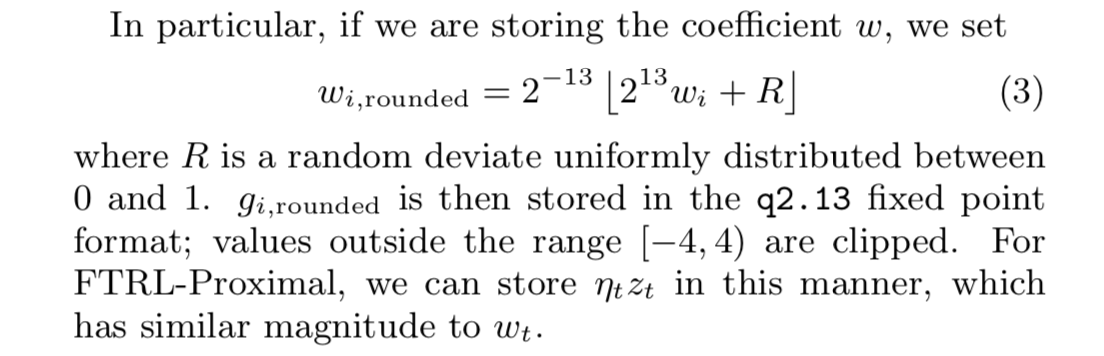
1. Poisson inclusion: 对于一个新特征我们以概率p的大小将这个特征加入模型。然后我们用OGD来更新相关的统计量和系数矩阵。
2. Bloom filter inclusion: we use a rolling set of counting bloom filters to detect the first n times a feature is encountered in training. Once a feature has occurred more than n times(according to the filter), we add it to the model and use it for training in subsequent observations as above. Note that this methods is also probabilistic, because a counting bloom filter is capable of false positives. That is, we will sometimes include a feature that has actually occurred less than n times.

4.2 encoding values with fewer bits

一般对于OGD我们用的是floating point encoding，但是这不够高效，我们现在用的一般是q2.13编码。对于一个16位的编码，按照以下规则：



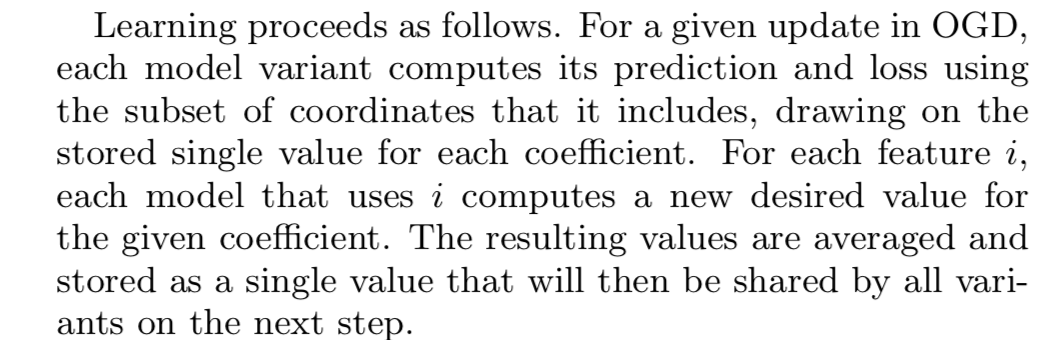
采用q2.13编码进行存储后，它的形式变为：



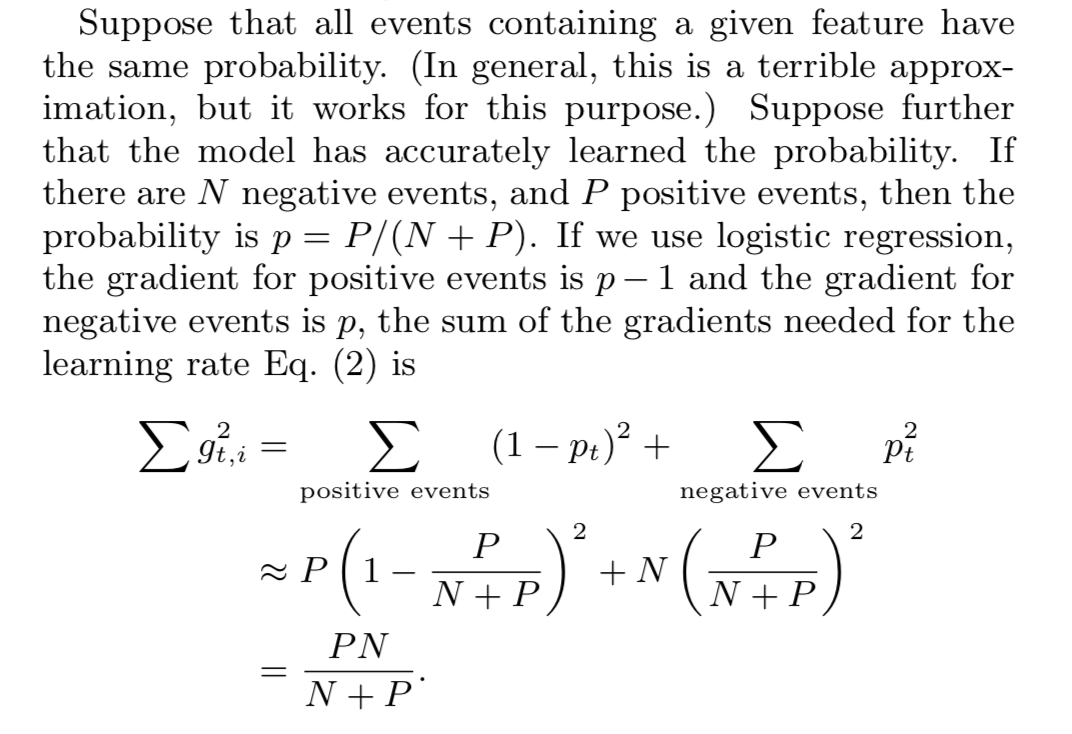
4.3 training many similar models

Since we train together only highly similar models, the memory saving from not representing the key and the memory saving from not representing the key and the counts per model is much larger than the loss from features not in common.

4.4 a single value structure

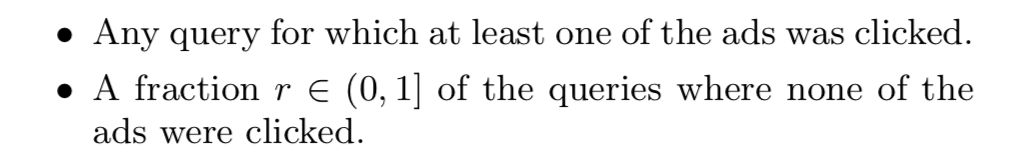


4.5 computing learning rates with counts

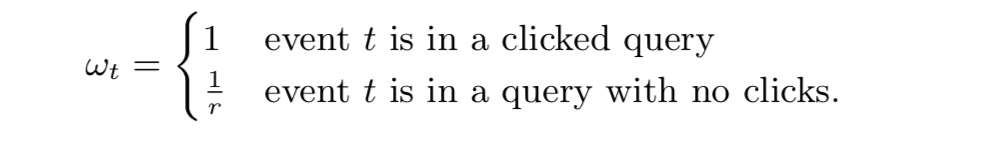


正例的梯度是1-p更对吧？

4.6 subsampling training data

’

对于click query和no click query取不同的权重：



Evaluation model performance

5.1 progressive validation

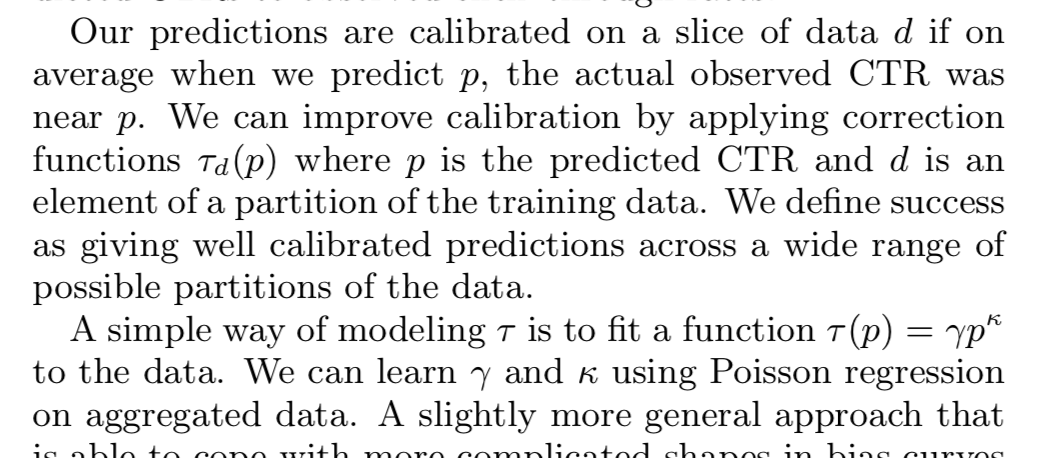
Progressive validation sometimes called online loss.

Look at the relative changes

5.2 deep understanding through visualization

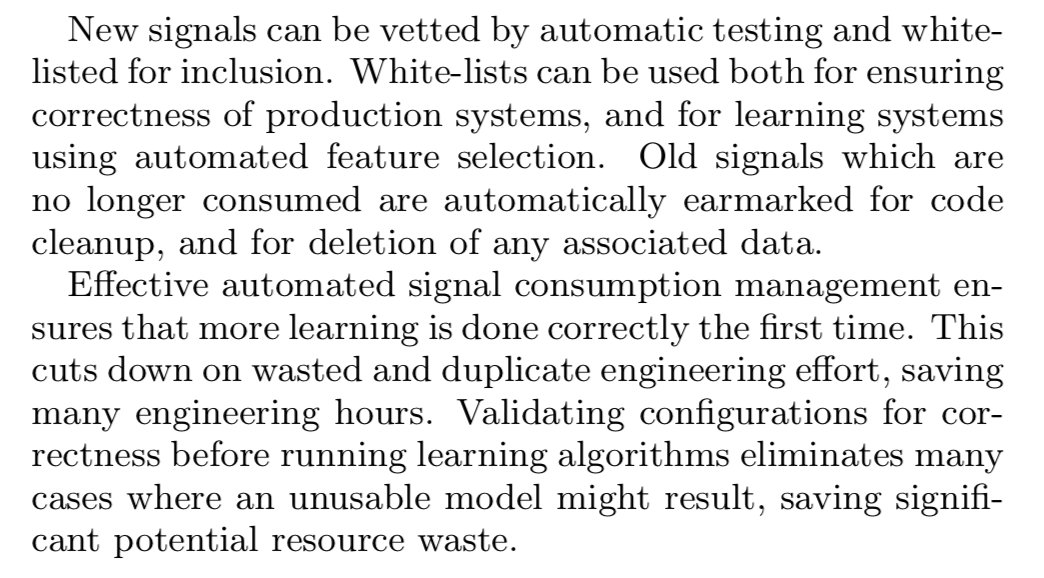
利用自定义的GridViz方法来查看

Confidence estimates

Calibrating predictions:  


Automated feature management:

When an improved version of a signal is made available, consumers can be alerted to experiment with the new version.



说了自动化特征管理的好处，但是没有具体介绍如何实现

Unsuccessful experiments

9.1 aggressive feature hashing

通过实验，发现对特征进行hash，对效果并没有提升

9.2 dropout

因为在我们的数据中input feature are sparse and labels are noisy. In our sparse, noisy setting adding in dropout appears to simply reduce the amount of data available for learning

9.3 feature bagging

9.4 feature vector normalization