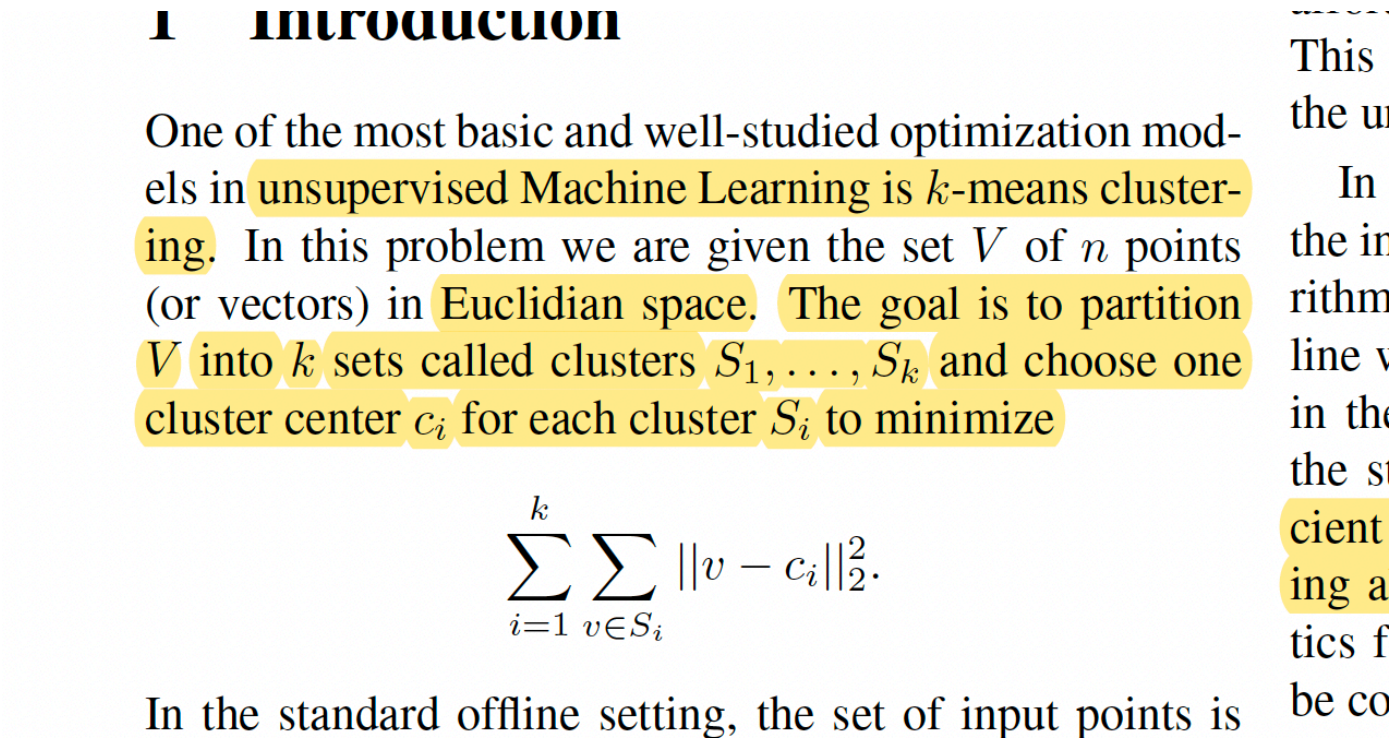
Typical model —— K-means

One of the most basic and well-studied optimization models in unsupervised Machine Learning is k-means clustering.

In this problem we are given the set V of n points (or vectors) in Euclidian space. The goal is to partition V into k sets called clusters S1; : : : ; Sk and choose one cluster center ci for each cluster Si to minimize. In the standard offline setting, the set of input points is



1 Introduction

One of the most basic and well-studied optimization models in unsupervised Machine Learning is k-means clustering. In this problem we are given the set V of n points (or vectors) in Euclidian space. The goal is to partition V into k sets called clusters $S_1, \ldots, S_k$ and choose one cluster center c_i for each cluster S_i to minimize

$$\sum_{i=1}^{k} \sum_{v \in S_i} ||v - c_i||_2^2.$$

In the standard offline setting, the set of input points is

This
the u
In
the in
rithm
line v
in the
the st
cient
ing a
tics f
be co

known in advance and the data access model is unrestricted. Even so, obtaining provably good solutions to this problem is difficult.

In the streaming model the algorithm must consume the data in one pass and is allowed to keep only a small amount of information. Nevertheless, it must output its final decisions when the stream has ended. In contrast, the online model of computation does not allow to postpone clustering decisions In this setting, an a priori unknown number of points arrive one by one in an arbitrary order. When a new point arrives the algorithm must either put it in one of the existingNote that this problem is conceptually non trivial even if one could afford unbounded computational power at every iteration. This is because the quality of current choices depend on the unknown (yet unseen) remainder of the stream.clusters or open a new cluster (consisting of a single point).

T**he** **specific problem to be addressed by this paper**

In the streaming model the algorithm must consume the data in one pass and is allowed to keep only a small amount of information. Nevertheless, it must output its final decisions when the stream has ended. In this setting, an a priori unknown number of points arrive one by one in

an arbitrary order. When a new point arrives the algorithm must either put it in one of the existingNote that this problem is conceptually non trivial even if one could afford unbounded computational power at every iteration. This is because the quality of current choices depend on the unknown (yet unseen) remainder of the stream.clusters or open a new cluster (consisting of a single point).

Online K-means (No prior knowledge)**

Semi-online K-means (Having a lower bound)**

In this work we provide algorithms for both online k-means and semi-online k-means. In the semi-online model we assume having a lower bound, w, for the total optimal cost of k-means, W, as well as an estimate for n, the length of the stream.

In the fully online model we do not assume any prior knowledge

1. Semi-Online k-means Algorithm

For the facility cost, start with f1 which is known to be too low. By doing that the algorithm is “encouraged” to open many facilities (centers) which keeps the service costs low. If the algorithm detect that too many facilities were opened, it can conclude that the current facility cost is too low. It therefore doubles the facility cost of opening future facilities (centers). It is easy to see that the facility cost cannot be doubled many times without making opening new clusters prohibitively expensive.

2.** Fully Online k-means Algorithm**

Algorithm 2 is fully online yet it defers from Algorithm 1 in only a few aspects. First, since n is unknown, the initial facility cost and the doubling condition cannot depend on it. Second, it must generate its own lower bound w based on a short prefix of points in the stream. Note that w is trivially smaller that W. Any clustering of k + 1 points must put at least two points in one cluster, incurring a cost of kv v0k2=2 minv;v0 kv v0k2=2.

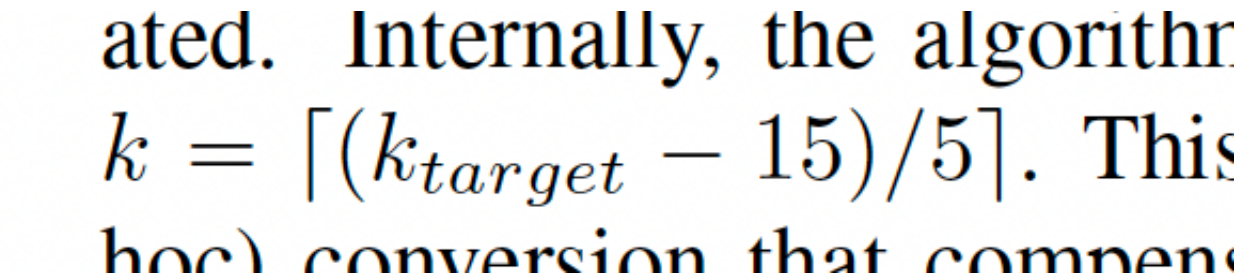
Algorithm 2 is fully online yet it defers from Algorithm 1 in only a few aspects. First, since n is unknown, the initial facility cost and the doubling condition cannot depend on it. Second, it must generate its own lower bound w^* based on a short prefix of points in the stream. Note that w^* is trivially smaller that W^*. Any clustering of $k+1$ points must put at least two points in one cluster, incurring a cost of $\|v - v'\|^2/2 \geq \min_{v,v'} \|v - v'\|^2/2$.

Theo
of Al
cost.

Proo
Theo

3**. modifications** **to the algorithm**

While experimenting with the algorithm, we discovered that some log factors were, in fact, too pessimistic in practice. We also had to make some pragmatic decisions about, for example, how to set the initial facility cost. As another practical adjustment we introduce the notion of ktarget and kactual. The value of ktarget is the number of clusters we would like the algorithm to output while kactual is the actual number of clusters generated. Internally, the algorithm operates with a value of k = d(ktarget 15)=5e. This is a heuristic (entirely adhoc) conversion that compensates for the kactual being larger than k by design.



Experimental —— Datasets on 12 different datasets 1

To evaluate our algorithm we executed it on 12 different datasets. All the datasets that we used are conveniently aggregated on the LibSvm website [14] and on the UCI dataset collection [7]. Some basic information about each dataset is given in Table 1.

Feature engineering for the sake of online learning is one of the motivations for this work. For that reason, we apply standard stochastic gradient descent linear learning with the squared loss on these data. Once with the raw features and once with the k-means features added. In some cases we see a small decrease in accuracy due to slower convergence of the learning on a larger feature set. This effect should theoretically be nullified in the presence of more data. In other cases, however, we see a significant uptick in classification accuracy.

Experimental —— Datasets on 12 different datasets 2

Feature engineering for the sake of online learning is one of the motivations for this work. For that reason, we apply standard stochastic gradient descent linear learning with the squared loss on these data. Once with the raw features and once with the k-means features added. In some cases we see a small decrease in accuracy due to slower convergence of the learning on a larger feature set. This effect should theoretically be nullified in the presence of more data. In other cases, however, we see a significant uptick in classification accuracy.

Corroborating the observations of [11] we report that adding k-means feature, particularly to low dimensional datasets, is very beneficial for improving classification accuracy. Indeed enabling online machine leaning to take advantage of this phenomenon is one of the motivations for performing k-means online.

Experimental —— Number of cluster | actual vs target

One of the artifacts of applying our online k-means algorithm is that the number of clusters is not exactly known a priory. But as we see in Figure 1, the number of resulting clusters is rather predictable and controllable. Figure 1 gives the ratio between the number of clusters output by the algorithm, kactual, and the specified target ktarget. The results reported are mean values of 3 runs for every parameter setting. The observed standard deviation of kactual is typically in the range [0; 3] and never exceeded 0:1 ktarget in any experiment. Figure 1 clearly shows that the ratio kactual=ktarget is roughly constant and close 1:0. Interestingly, the main differentiator is the choice of dataset.

Experimental —— Online k-means clustering cost (f-online) as a **function** of** **k**-target** **for the different datasets**.**

Throughout this section, we measure the online k-means clustering cost with respect to different baselines. We report averages of at least 3 different independent execu-tions for every parameter setting. In Figure 2 the reader can see the online k-means clustering cost for the set of centers chosen online by our algorithm for different values of ktarget and different datasets. For normalization, each cost is divided by f0, the sum of squares of all vector norms in the dataset (akin to the theoretical k-means cost of having one center at the origin). Note that some datasets are inherently unclusterable. Even using many cluster centers, the k-means objective does not decrease substantially. Nevertheless, as expected, the k-means cost obtained by the online algorithm, fonline, decreases as a function of ktarget.

Experimental —— Online k-means clustering cost (f-online) as a **function** of** **k**-target** **for the different datasets**.**

The next experiment compares online k-means to k-means++. For every value of ktarget we ran online k-means to obtain both fonline and kactual. Then, we invoke k-means++ using kactual clusters and computed its cost, fkmpp. This experiment was repeated 3 times for each dataset and each value of ktarget. The mean results are reported in Figure 4.

means++. For every value of k_{target} we ran online k-means to obtain both f_{online} and k_{actual}. Then, we invoke k-means++ using k_{actual} clusters and computed its cost, f_{kmpp}. This experiment was repeated 3 times for each dataset and each value of k_{target}. The mean results are reported in Figure 4. Unsurprisingly, k-means++ is usually better in terms of cost. But, the reader should keep in mind that k-means++ is an *offline* algorithm that requires k passes over the data compared with the online computational model of our algorithm.

Summarize

Unsurprisingly, k-means++ is usually better in terms of cost. But, the reader should keep in mind that k-means++ is an offline algorithm that requires k passes over the data compared with the online computational model of our algorithm.