An Algorithm for Online K-Means Clustering

Edo Liberty[*] Ram Sriharsha[†] Maxim Sviridenko[‡]

Abstract

This paper shows that one can be competitive with the k-means objective while operating online. In this model, the algorithm receives vectors $v_1, \ldots, v_n$ one by one in an arbitrary order. For each vector v_t the algorithm outputs a cluster identifier before receiving v_{t+1}. Our online algorithm generates $O(k \log n \log \gamma n)$ clusters whose expected k-means cost is $O(W^* \log n)$. Here, W^* is the optimal k-means cost using k clusters and γ is the aspect ratio of the data. The dependence on γ is shown to be unavoidable and tight. We also show that, experimentally, it is not much worse than k-means++ while operating in a strictly more constrained computational model.

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 $S_1, \ldots, S_k$ to minimize

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

where $c_i = (1/|S_i|) \sum_{v \in S_i} v$ are the centers of the clusters S_i. Alternatively, one could compute the centers c_i and define the clusters S_i as the set of points for which c_i is the closest center. While the two formulation are equivalent, which one is more natural and what algorithms are applicable very much depend on the data access model.

Offline: In the standard offline setting, the set of input points is known in advance and the data access model is unrestricted. Even so, obtaining provably good solutions to this problem is difficult. Lloyd's algorithm [20] provides a popular and powerful heuristic. It is so popular that practitioners often simply refer to it as k-means. Yet, only recently some theoretical guaranties were proven for its performance on "well clusterable" inputs [23]. The k-means++ [5] algorithm provides an expected $O(\log(k))$ approximation or an efficient seeding algorithm. A well known theoretical algorithm is due to Kanungo et al. [18]. It gives a constant approximation ratio and is based on local search ideas popular in the related area of design and analysis of algorithms for facility location problems, e.g., [6]. Recently, [2] improved the analysis of [5] and gave an adaptive sampling based algorithm with constant factor approximation to the optimal cost. In an effort to make adaptive sampling techniques more scalable, [8] introduced k-means$\|$ which reduces the number of passes needed over the data and enables improved parallelization.

Streaming: In the streaming model the algorithm must consume the data in one pass and is allowed to keep only a small (typically constant or poly-logarithmic in n) amount of information. Nevertheless, it must output cluster centers (c_i) when the stream terminates. This severely restricted data access model requires new algorithmic ideas. The streaming model was considered by [3] and [22] and later by [1]. They build upon adaptive sampling ideas from [5, 8] and divide-and-conquer techniques from [17]. Note that, in this model, the assignment of individual points to clusters becomes available only in hindsight.

Online: In contrast, an online k-means algorithm must assign points to clusters throughout the run of the algorithm. 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 add it in one of the existing clusters or open a new cluster (consisting of a single point).

This setting is strictly harder than the streaming model. On the one hand, any *space efficient* online algorithm is trivially convertible to a streaming algorithm. One could keep sufficient statistics for each cluster such that the centers of mass could be computed at the end of the stream. The converse is not true. The online problem is conceptually non trivial even if one could afford unbounded memory and computational power at every iteration. This is because the utilities of current choices depend on the unknown (yet unseen) remainder of the stream.

Consider the stream of one-dimensional vectors $1, c, c^2, \ldots, c^n$ and $k = 2$ and assume for convenience

[*] edo@yahoo-inc.com, Yahoo Labs, New York, NY

[†] harshars@yahoo-inc.com, Yahoo Labs, Sunnyvale, CA

[‡] sviri@yahoo-inc.com,Yahoo Labs, New York, NY

that $c \geq 2$. At time t the online algorithm observes the value c^t. If the algorithm appends this point to an existing cluster, it incurs a cost of at least $c^{2t}/8$. The best offline clustering would assign $1, c, c^2, ...c^{t-1}$ to one cluster and c^t to another. Thereby incurring a total k-means cost of at most $2c^{2(t-1)}$. If the algorithm does not create a new singleton cluster for c^t it commits to an $\Omega(c^2)$ approximation ratio. Therefore, *any* online algorithm for k means must suffer either an unbounded approximation ratio or create an unbounded number clusters.

1.1 Motivations for online clustering

In the context of machine learning, the results of k-means were shown to provide powerful unsupervised features [11] on par, sometimes, with neural nets for example. This is often referred to as (unsupervised) feature learning. Intuitively, if the clustering captures most of the variability in the data, assigning a single label to an entire cluster should be pretty accurate. It is not surprising therefore that cluster labels are powerful features for classification. In the case of online machine learning, these cluster labels must also be assigned online. The importance of such an online k-means model was already recognized in machine learning community [10, 12].

For information retrieval, [9] investigated the incremental k-centers problem. They argue that clustering algorithms, in practice, are often required to be online. We observe the same at Yahoo. For example, when suggesting news stories to users, we want to avoid suggesting those that are close variants of those they already read. Or, conversely, we want to suggest stories which are a part of a story-line the user is following. In either scenario, when Yahoo receives a news item, it must immediately decide what cluster it belongs to and act accordingly.

1.2 Our contributions

In this work we show that there exists an online k-means algorithm (Algorithm 2) such that a) its expected cost is $O(W^* \log n)$ and b) it creates at most $O(k \log n \log \gamma n)$ clusters in expectation. Here, and throughout, W^* denotes the best offline cost and $\gamma = \max_{v,v'} \|v - v'\| / \min_{v,v'} \|v - v'\|$ is the dataset "aspect ratio". To the best of our knowledge, we present the first online algorithm for k-means with worst case bounds.

The hard example in the introduction (vectors $1, c, c^2, \ldots, c^n$) required n clusters and exhibits aspect ratio $\gamma = O(c^n)$. It shows that a logarithmic dependence between the number of clusters and the aspect ratio is necessary. Algorithm 2 completes the picture by showing that it is also sufficient.

We also give a result for the semi-online setting (Algorithm 1). In reality, one often has a priori rough estimates for both n and W^*. We show that, even if these estimates are wrong up to factors of $n^{O(1)}$ there still exists an online algorithm such that a) its expected cost is $O(W^*)$ and b) it creates at most $O(k \log^2 n)$ clusters in expectation.

Experimentally, we observe the known phenomenon that features derived from k-means clustering are powerful features for classification tasks. We extend this observation and show that *online* k-means is useful for *online* learning. We also show that the clustering cost of our online algorithm is comparable to that of k-means++. That, even though it operates in much more restricted computational setting.

1.3 Prior Art

The first (to our knowledge) result in online clustering dates back the k-centers result of [9]. For k-means an Expectation Maximization (EM) approach was investigated by [19]. Their focus was on online EM as a whole but their techniques include online clustering. They offer very encouraging results, especially in the context of machine learning. To the best of our understanding, however, their techniques do not extend to arbitrary input sequences. In contrast, the result of [10] provides provable results for the online setting in the presence of base-k-means algorithm as experts.

A closely related family of optimization problems is known as facility location problems. Two standard variants are the uncapacitated facility location problem (or the simple plant location problem in the Operations Research jargon) and the k-median problem. These problems are well-studied both from computational and theoretic viewpoints (a book [13] and a survey [24] provide the background on some of the aspects in this area). Meyerson [21] suggested a simple and elegant algorithm for the online uncapacitated facility location with competitive ratio of $O(\log n)$. Fotakis [15] suggested a primal-dual algorithm with better performance guarantee of $O(\log n / \log \log n)$. Anagnostopoulos et al. [4] considered a different set of algorithms based on hierarchical partitioning of the space and obtained similar competitive ratios. The survey [16] summarizes the results in this area. As a remark, [9] already considered connections between facility location problems and clustering. Interestingly, their algorithm is often referred to as "the doubling algorithm" since the cluster diameters double as the algorithm receives more points. In our work the facility location cost is doubled which is technically different but intuitively related.

2 Semi-Online k-means Algorithm

We begin with presenting the *semi-online* algorithm. It assumes knowing the number of vectors n and some lower bound w^* for the value of the optimal solution. These assumptions make the algorithm slightly simpler than Algorithm 2. Nevertheless, the *semi-online* online already faces most of the challenges faced by the *fully online* version. In fact, proving the correctness of the online algorithm (Section 3) would require only minor adjustments to the proofs in this section.

The algorithm uses ideas from the online facility location algorithm of Meyerson [21]. The intuition is as follows; think about k-means and a facility location problem where the service costs are squared Euclidean distances. Consider starting with a facility cost that is "too low" which means that the algorithm is "encouraged" to open many facilities (centers). This keeps the service costs low. Obtaining an initial "too low" facility cost is explained below. If the algorithm detects that too many facilities were opened, it can conclude that the current facility cost is indeed too low. The algorithm then 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 the cost of opening new facilities prohibitively expensive. In Algorithm 2 we denote the distance of a point v to a set C as $D(v, C) = \min_{c \in C} \|v - c\|$. As a convention, if $C = \emptyset$ then $D(v, C) = \infty$ for any v.

Algorithm 1 semi-online k-means algorithm

```
input: $V$, $k$, $w^*$, $n$
$C \leftarrow \emptyset$
$r \leftarrow 1$; $q_1 \leftarrow 0$; $f_1 \leftarrow w^*/k\log(n)$
for $v \in V$ do
  with probability $p = \min(D^2(v, C)/f_r, 1)$
        $C \leftarrow C \cup \{v\}; q_r \leftarrow q_r + 1$
  if $q_r \geq 3k(1 + \log(n))$ then
    $r \leftarrow r + 1$; $q_r \leftarrow 0$; $f_r \leftarrow 2 \cdot f_{r-1}$
  end if
  yield: $c = \arg\min_{c \in C} \|v - c\|^2$
end for
```

Consider some optimal solution consisting of clusters $S_1^*, \ldots, S_k^*$ with cluster centers $c_1^*, \ldots, c_k^*$. Let

$$W_i^* = \sum_{v \in S_i^*} ||v - c_i^*||_2^2$$

be the cost of the i-th cluster in the optimal solution and $W^* = \sum_{i=1}^k W_i^*$ be the value of the optimal solution. Let A_i^* be the average squared distance to the cluster center from a vectors in the i-th optimal cluster.

$$A_i^* = \frac{1}{|S_i^*|} \sum_{v \in C_i^*} \|v - c_i^*\|_2^2 = \frac{W_i^*}{|S_i^*|} \ .$$

We define a partition of the cluster S_i^* into subsets that we call *rings*:

$$S_{i,0}^* = \{v \in S_i^* : ||v - c_i^*||_2^2 \leq A_i^*\}$$

and for $1 \leq \tau \leq \log n$

$$S_{i,\tau}^* = \left\{v \in S_i^* : ||v - c_i^*||_2^2 \in (2^{\tau-1}A_i^*, 2^\tau A_i^*]\right\} \ .$$

Note that we consider only values of $\tau \leq \log n$ since $S_{i,\tau}^* = \emptyset$ for $\tau > \log(|S_i^*|)$. To verify assume the contrary and compute A_i^*.

Theorem 2.1. *Let C be the set of clusters defined by Algorithm 1. Then*

$$\mathbb{E}[|C|] = O\left(k \log n \log \frac{W^*}{w^*}\right) \ .$$

Proof. Consider the phase r' of the algorithm where, for the first time

$$f_{r'} \geq \frac{W^*}{k \log n} \ .$$

The initial facility cost f_1 is doubled at every phase during each of which the algorithm creates $3k(1+\log n)$ clusters. The total number of clusters opened before phase r' is upper bounded by $3k(1 + \log n) \log \frac{f_{r'}}{f_1}$. Which is, in turn, $O(k \log n \log \frac{W^*}{w^*})$ by the choice of f_1.

Bounding the number of centers opened during and after phase r' is more complicated. Denote by $S_{i,\tau,r}^*$ the set of points in the ring $S_{i,\tau}^*$ that our algorithm encounters during phase r. The expected number of clusters initiated by vectors in the ring $S_{i,\tau}^*$ during phases $r', \ldots, R$ is at most

$$1 + \sum_{r \geq r'} \frac{4 \cdot 2^\tau A_i^*}{f_r} |S_{i,\tau,r}^*| \ .$$

This is because once we open the first cluster with a center at some $v \in S_{i,\tau}^*$ the probability of opening a cluster for each subsequent vector $v' \in S_{i,\tau}^*$ is upper bounded by

$$\frac{||v - v'||_2^2}{f_r} \leq \frac{2||v - c_i^*||_2^2 + 2||v' - c_i^*||_2^2}{f_r} \leq \frac{4 \cdot 2^\tau A_i^*}{f_r}$$

by the (squared) triangle inequality for $v, v' \in S_{i,\tau}^*$.

Therefore the expected number of cluster centers chosen from S_i^* during and after phase r' is at most

$$\begin{aligned}
&\sum_{\tau\ge 0}\left(1+\sum_{r\ge r'}\frac{4\cdot 2^\tau A_i^*}{f_r}|S^*_{i,\tau,r}|\right)\\
&\le 1+\log n+\sum_{\tau\ge 0}4\cdot 2^\tau A_i^*\sum_{r\ge r'}\frac{|S^*_{i,\tau,r}|}{f_r}\\
&\le 1+\log n+\frac{4}{f_{r'}}\sum_{\tau\ge 0}2^\tau A_i^*|S^*_{i,\tau}|\\
&\le 1+\log n+\frac{4}{f_{r'}}A_i^*|S^*_{i,0}|+\frac{8}{f_{r'}}\sum_{\tau\ge 1}2^{\tau-1}A_i^*|S^*_{i,\tau}|\\
&\le 1+\log n+\frac{4}{f_{r'}}A_i^*|S_i^*|+\frac{8}{f_{r'}}\sum_{\tau\ge 1}\sum_{v\in S^*_{i,\tau}}\|v-c_i^*\|^2\\
&\le 1+\log n+\frac{4}{f_{r'}}W_i^*+\frac{8}{f_{r'}}W_i^*\\
&\le 1+\log n+\frac{12W_i^*}{f_{r'}}.
\end{aligned}$$

Summing up over all $i=1,\ldots,k$ using $\sum_i W_i^*=W^*$ we obtain that the expected number of centers chosen during phases $r',\ldots,R$ is at most

$$k(1+\log n)+12W^*/f_{r'}\ . \tag{2.1}$$

Substituting $f_{r'}\ge W^*/k\log n$ completes the proof of the theorem.

Before we estimate the expected cost of clusters opened by our online algorithm we prove the following technical lemma.

Lemma 2.1. *We are given a sequence $X_1,\ldots,X_n$ of n independent experiments. Each experiment succeeds with probability $p_i\ge\min\{A_i/B,1\}$ where $B\ge 0$ and $A_i\ge 0$ for $i=1,\ldots,n$. Let t be the (random) number of consecutive unsuccessful experiments before the first successful one, then:*

$$\mathbb{E}\left[\sum_{i=1}^{t}A_i\right]\le B.$$

Proof. Let n' be the maximal index for which $p_i<1$ for all $i\le n'$.

$$\begin{aligned}
\mathbb{E}\left[\sum_{i=1}^{t}A_i\right] &= \sum_{i=1}^{n'}A_i\Pr[t\ge i]\\
&\le \sum_{i=1}^{n'}A_i\prod_{j=1}^{i}\left(1-\frac{A_j}{B}\right)\\
&\le B\sum_{i=1}^{n'}\frac{A_i}{B}\prod_{j=1}^{i-1}\left(1-\frac{A_j}{B}\right)\\
&\le B\sum_{i=1}^{n'}q_i\prod_{j=1}^{i-1}(1-q_j)\\
&\le B.
\end{aligned}$$

For the last transition, let $q_i=A_i/B\le p_i<1$ for $i\le n'$ be the probabilities of events. Then, $q_i\prod_{j=1}^{i-1}(1-q_j)$ is the probability that i is the first events that happens. Since these are mutually exlusive events $\sum_{i=1}^{n'}q_i\prod_{j=1}^{i-1}(1-q_j)\le 1$.

Theorem 2.2. *Let W be the cost of the online assignments of Algorithm 1 and W^* the optimal k-means clustering cost. Then*

$$\mathbb{E}[W]=O(W^*)\ .$$

Proof. Consider the service cost of vectors in each ring $S^*_{i,\tau}$ in two separate stages. Before a vector from the ring is chosen to start a new cluster and after. Before a center from $S^*_{i,\tau}$ is chosen each vector $v\in S^*_{i,\tau}$ is chosen with probability $p\ge\min\{d^2(v,C)/f_R,1\}$ where R is the final value of R in the algorithm. Here, C is the set of centers already chosen by the algorithm before encountering v. If v is not chosen the algorithm incurs a cost of $d^2(v,C)$. By Lemma 2.1 the expected sum of these costs is bounded by f_R. Summing over all the rings we get a contribution of $O(f_R k\log n)$.

After a vector $v\in S^*_{i,\tau}$ is chosen to start a new cluster, the service cost of each additional vector v' is at most $\|v-v'\|^2\le 4\cdot 2^\tau A_i^*$. Summing up over all vectors and rings, this stage contributes are most $4\sum_i\sum_\tau\cdot 2^\tau A_i^*\cdot|S^*_{i,\tau}|\le 12W^*$ to the cost of our solution. All in all, the expected online k-means cost is bounded by

$$\mathbb{E}[W]=O(f_R k\log n+W^*)\ .$$

We now turn to estimating $\mathbb{E}[f_R]$. Consider the first phase r'' of the algorithm such that

$$f_{r''}\ge\frac{36W^*}{k(1+\log n)}\ .$$

By Equation 2.1 the expected number of clusters opened during and after phase r'' is at most $k(1+\log n)+12W^*/f_{r''}\le\frac{4}{3}k(1+\log n)$. By Markov's inequality the probability of opening more than $3k(1+\log n)$ clusters is at most $4/9$. Therefore, with probability at least $5/9$ the algorithm will conclude while at phase r''.

Let p be the probability that our algorithm terminates before round r''. Since the probability of concluding the execution at a round after r'' is at least $5/9$ we derive an upper bound

$$\begin{aligned}\mathbb{E}[f_R] &\le pf_{r''-1} + (1-p)\sum_{r=r''}^{+\infty} f_r \cdot \frac{5}{9} \cdot \left(\frac{4}{9}\right)^{r-r''} \\ &< f_{r''} + f_{r''} \cdot \frac{5}{9}\sum_{i=0}^{+\infty} 2^i \cdot \left(\frac{4}{9}\right)^i = O(f_{r''})\end{aligned}$$

Combining $\mathbb{E}[f_R] = O(f_{r''})$ with our choice of $f_{r''} = O(\frac{W^*}{k(1+\log n)})$ and our previous observation that $\mathbb{E}[W] = O(f_R k \log n + W^*)$ completes the proof.

3 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 smaller that W^*. Any clustering of $k+1$ points must put at least two points in one cluster, incurring a cost of $\|v - v'\|^2/2 \ge \min_{v,v'} \|v - v'\|^2/2$.

Algorithm 2 Online k-means algorithm

```
  input: $V$, $k$
  $C \leftarrow$ first $k+1$ distinct vectors in $V$; and $n = k+1$
  (For each of these yield itself as its center)
  $w^* \leftarrow \min_{v,v' \in C} \|v - v'\|^2/2$
  $r \leftarrow 1$; $q_1 \leftarrow 0$; $f_1 = w^*/k$
  for $v \in$ the remainder of $V$ do
    $n \leftarrow n + 1$
    with probability $p = \min(D^2(v, C)/f_r, 1)$
          $C \leftarrow C \cup \{v\}$; $q_r \leftarrow q_r + 1$
    if $q_r \ge 3k(1 + \log(n))$ then
      $r \leftarrow r + 1$; $q_r \leftarrow 0$; $f_r \leftarrow 2 \cdot f_{r-1}$
    end if
    yield: $c = \arg\min_{c \in C} \|v - c\|^2$
  end for
```

Theorem 3.1. *Let C be the set of clusters defined by Algorithm 2. Then*

$$\mathbb{E}[|C|] = O\left(k \log n \log \frac{W^*}{w^*}\right) = O\left(k \log n \log \gamma n\right) \ .$$

Here $\gamma = \frac{\max_{v,v'} \|v-v'\|}{\min_{v,v'} \|v-v'\|}$ is the dataset "aspect ratio".

Proof. Intuitively, for the same lower bound w^* Algorithm 2 should create fewer centers than Algorithm 1 since its initial facility cost is higher and it is doubled more frequently. This intuition is made concrete by retracing the proof of Theorem 3.1 to show

$$\mathbb{E}[|C|] = O\left(k \log n \log \frac{W^*}{w^*}\right) \ .$$

To get a handle on the value of W^*/w^*, observe that $W^* \le n \max_{v,v'} \|v - v'\|^2$. Combining this with the definition of γ we get $\log(W^*/w^*) = O(\log \gamma n)$.

Theorem 3.2. *Let W be the cost of the online assignments of Algorithm 2 and W^* the optimal k-means clustering cost. Then*

$$\mathbb{E}[W] = O(W^* \log n) \ .$$

Proof. We start by following the argument of the proof of Theorem 2.2 verbatim. We arrive at the conclusion that

$$\mathbb{E}[W] = O(f_R k \log n + W^*)$$

where f_R is the final facility cost of the algorithm and R is its last phase. Showing that $\mathbb{E}[f_R] = O(W^*/k)$ will therefore complete the proof.

Consider any phase $r \ge r''$ of the algorithm where r'' is the smallest index such that

$$f_{r''} \ge \frac{36W^*}{k}.$$

Let n_r be the number of points from the input the algorithm went through by the end of phase r. Let q_r be the number of clusters opened during phase r and q'_r the number those who are *not* the first in their ring.

$$q_r \le k \log(1 + \log n_r) + q'_r$$

The term $k \log(1 + \log n_r)$ is an upper bound on the number of rings at the end of stage r. We pessimistically count at most one (the first) cluster from each such ring. Following the argument in the proof of Theorem 2.1 that lead us to Equation (2.1) we conclude $\mathbb{E}[q'_r] \le 12W^*/f_r$.

Algorithm 2 only advances to the next phase if $q_r \ge 3\log(1+\log n_r)$ which requires $q'_r \ge 2k(1+\log n_r)$. By Markov's inequality and the fact that $\mathbb{E}[q'_r] \le 12W^*/f_r \le k/3$ the probability of reaching the next phase is at most $1/6$.

We now estimate $\mathbb{E}[f_R]$. Let p be the probability that our algorithm finishes before round r''. We have

$$\begin{aligned}\mathbb{E}[f_R] &\le pf_{r''-1} + (1-p)\sum_{r=r''}^{+\infty} f_r \cdot \frac{5}{6} \cdot \left(\frac{1}{6}\right)^{r-r''} \\ &\le f_{r''} + f_{r''} \cdot \frac{5}{6}\sum_{i=0}^{+\infty} 2^i \cdot \left(\frac{1}{6}\right)^i = O(f_{r''})\end{aligned}$$

Since $f_{r''} = O(W^*/k)$ the proof is complete.

4 Experimental Analysis of the Algorithm

4.1 Practical 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 k_{target} and k_{actual}. The value of k_{target} is the number of clusters we would like the algorithm to output while k_{actual} is the actual number of clusters generated. Internally, the algorithm operates with a value of $k = \lceil (k_{target} - 15)/5 \rceil$. This is a heuristic (entirely ad-hoc) conversion that compensates for the k_{actual} being larger than k by design. Nevertheless, Algorithm 3 is nothing but a practical variant of Algorithm 2. It is only detailed below to support reproduction of our results.

Algorithm 3 Online k-means algorithm - experimental

```
input: $V$, $k_{target}$
$k = \lceil (k_{target} - 15)/5 \rceil$
$C \leftarrow$ the first $k + 10$ vectors in $V$
(For each of these yield itself as its center)
$w^* \leftarrow$ half the sum of the 10 smallest squared distances of points in $C$ to their closest neighbor.
$r \leftarrow 1$; $q_1 \leftarrow 0$; $f_1 \leftarrow w^*$
for $v \in$ the remainder of $V$ do
  with probability $p = \min(D^2(v, C)/f_r, 1)$
      $C \leftarrow C \cup \{v\}$; $q_r \leftarrow q_r + 1$
  if $q_r \geq k$ then
    $r \leftarrow r + 1$; $q_r \leftarrow 0$; $f_r \leftarrow 10 \cdot f_{r-1}$
  end if
  yield: $c = \arg\min_{c \in C} \|v - c\|^2$
end for
$k_{actual} \leftarrow |C|$
```

4.2 Datasets 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 online learning (stochastic gradient descent with the squared loss) to several standard datasets. This is performed once with the raw features and once with the online 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 negligible in the presence of more data. In other cases, however, we see a significant uptick in classification accuracy. This is in agreement with prior observations [11].

Dataset	nnz	n	d
20news-binary	2.44E+6	1.88E+4	6.12E+4
adult	5.86E+5	4.88E+4	1.04E+2
ijcnn1	3.22E+5	2.50E+4	2.10E+1
letter	2.94E+5	2.00E+4	1.50E+1
magic04	1.71E+5	1.90E+4	9.00E+0
maptaskcoref	6.41E+6	1.59E+5	5.94E+3
nomao	2.84E+6	3.45E+4	1.73E+2
poker	8.52E+6	9.47E+5	9.00E+0
shuttle	2.90E+5	4.35E+4	8.00E+0
skin	4.84E+5	2.45E+5	2.00E+0
vehv2binary	1.45E+7	2.99E+5	1.04E+2
w8all	7.54E+5	5.92E+4	2.99E+2

Table 1: The table gives some basic information about the datasets we experimented with. The column under nnz gives the number of non zero entries in the entire dataset, n the number of vectors and d their dimension. Much more information is provided on LibSvm website [14] and in the UCI dataset collection [7].

Dataset	Classification accuracy with raw features	Classification accuracy with k-means features
20news	0.9532	0.9510
adult	0.8527	0.8721
ijcnn1	0.9167	0.9405
letter	0.7581	0.7485
magic04	1.0000	1.0000
maptaskcoref	0.8894	0.8955
nomao	0.5846	0.5893
poker	0.5436	0.6209
shuttle	0.9247	0.9973
skin	0.9247	0.9988
vehv2binary	0.9666	0.9645
w8all	0.9638	0.9635

Table 2: The work of [11] report that adding k-means feature, particularly to low dimensional datasets, is beneficial for improving classification accuracy. Here we report that this observation also hold analogously online. That is, online clustering features improve online classification accuarcy.

4.3 The number of online clusters One of the artifacts of applying our online k-means algorithm is that the number of clusters is not exactly known a priory. However, we see in Figure 1 that the number of resulting clusters is rather predictable and controllable. Figure 1 gives the ratio between the number of clusters output by the algorithm, k_{actual}, and the specified target k_{target}. The results reported are mean values of 3 runs for every parameter setting. The observed standard deviation of k_{actual} is typically in the range $[0, 3]$ and never exceeded $0.1 \cdot k_{target}$ in any experiment. Figure 1 clearly shows that the ratio k_{actual}/k_{target} is roughly constant and close 1.0. Interestingly, the main differentiator is the choice of dataset.

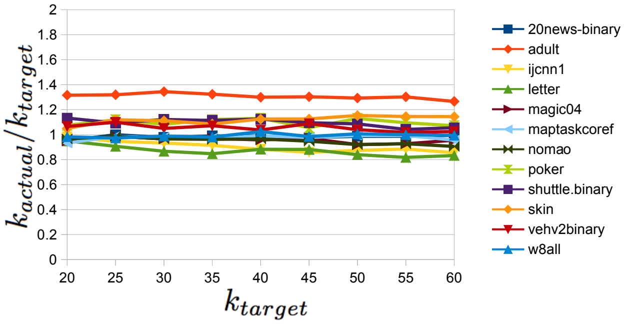

Figure 1: The figure gives the ratio k_{actual}/k_{target} on the y-axis as a function of k_{target} on the x-axis. The value k_{target} is given to the algorithm as input and k_{actual} is the resulting cardinality of the center set C. We clearly see that this ratio is roughly constant and close 1. Interestingly, the main differentiator is the dataset itself and not the value of k_{target}.

4.4 Online clustering cost 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 executions 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 k_{target} and different datasets. For normalization, each cost is divided by f_0, 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, f_{online}, decreases as a function of k_{target}.

The monotonicity of f_{online} with respect to k_{target} is unsurprising. In Figure 3 we plot the ratio f_{online}/f_{random} as a function of k_{target}. Here, f_{random} is the sum of squared distances of input points to k_{actual} input points chosen uniformly at random (as centers). Note that in each experiment the number of clusters used by the random solution and online k-means is identical, namely, k_{actual}. Figure 3 illustrates something surprising. The ratio between the costs remains relatively fixed per dataset and almost independent to k_{target}. Put differently, even when the k-means cost is significantly lower than picking k random centers, they improve in similar rates as k grows.

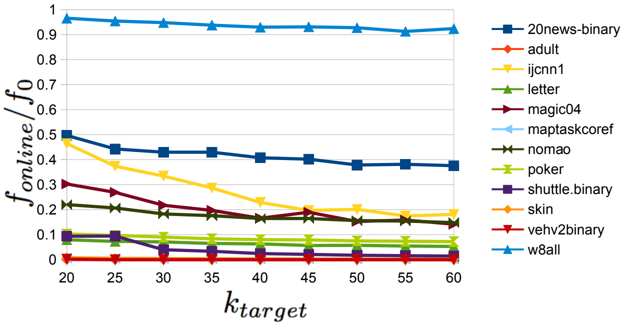

Figure 2: Online k-means clustering cost (f_{online}) as a function of k_{target} for the different datasets. For normalization, each cost is divided by f_0, the sum of squares of all vector norms in the dataset (akin to the k-cost of once center in the origin).

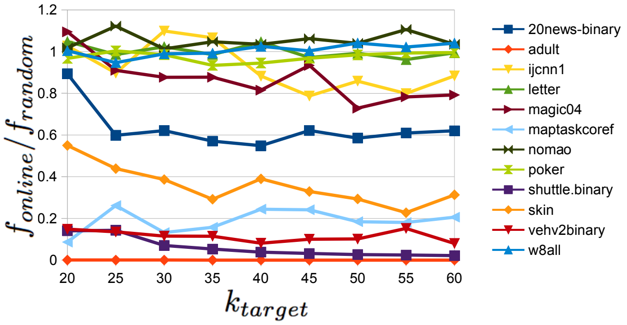

Figure 3: On the y-axis, the value of f_{online} divided by f_{random}. The latter is the cost of choosing, uniformly at random, as many cluster centers (from the data) as the online algorithm did. A surprising observation is that this ratio is almost constant for each dataset and almost independent of k_{target} (on the x-axis).

The next experiment compares online k-means to k-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. However, 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.

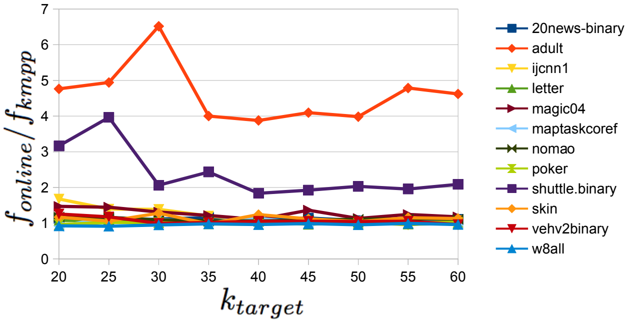

Figure 4: On the y-axis we plot f_{online}/f_{kmpp} as a function of k_{target} on the x-axis. The values of f_{online} is the cost of running Algorithm 3 with parameter k_{target}. The value of f_{kmpp} is the cost of running k-means++ with k_{actual} clusters, k_{actual} is the number of clusters online k-means actually used. We see that, except for the datasets *adult* and *shuttle.binary*, k-means++ and online k-means are comparable. For *adult* and *shuttle.binary* online k-means is worse by a small constant factor. Note (Figure 3) that both *adult* and *shuttle.binary* are datasets for which online k means is dramatically better than random.

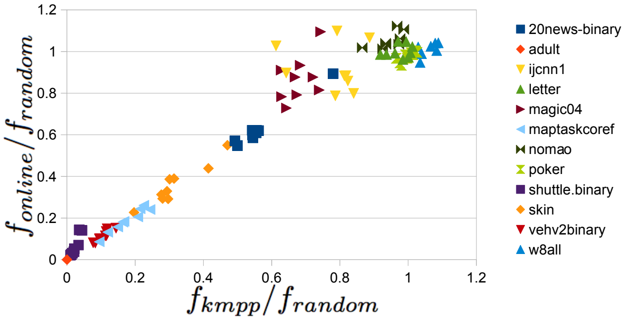

Figure 5: On the x-axis f_{kmpp}/f_{random} both using k_{actual} clusters. The value of k_{actual} is obtained by running online k-means with input k_{target} on the x-axis. The y-axis depicts f_{online}/f_{random}. Note that the performance of k-means++ and online k-means are very similar almost everywhere. The advantage of k-means++ (see Figure 4) occurs when the clustering cost is a minuscule fraction of random clustering.

5 Aknowledgements

We would like to thank Anna Choromanska and Sergei Vassilvitskii for very helpful suggestions and to Dean Foster for helping us with the proof of the Lemma 2.1.

References

[1] Marcel R. Ackermann, Marcus Märtens, Christoph Raupach, Kamil Swierkot, Christiane Lammersen, and Christian Sohler. Streamkm++: A clustering algorithm for data streams. *ACM Journal of Experimental Algorithmics*, 17(1), 2012.

[2] Ankit Aggarwal, Amit Deshpande, and Ravi Kannan. Adaptive sampling for k-means clustering. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 12th International Workshop, APPROX 2009, and 13th International Workshop, RANDOM 2009, Berkeley, CA, USA, August 21-23, 2009. Proceedings*, pages 15–28, 2009.

[3] Nir Ailon, Ragesh Jaiswal, and Claire Monteleoni. Streaming k-means approximation. In Yoshua Bengio, Dale Schuurmans, John D. Lafferty, Christopher K. I. Williams, and Aron Culotta, editors, *NIPS*, pages 10–18. Curran Associates, Inc., 2009.

[4] Aris Anagnostopoulos, Russell Bent, Eli Upfal, and Pascal Van Hentenryck. A simple and deterministic competitive algorithm for online facility location. *Inf. Comput.*, 194(2):175–202, 2004.

[5] David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In Nikhil Bansal, Kirk Pruhs, and Clifford Stein, editors, *SODA*, pages 1027–1035. SIAM, 2007.

[6] Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM J. Comput.*, 33(3):544–562, 2004.

[7] Kevin Bache and Moshe Lichman. UCI machine learning repository, 2013.

[8] Bahman Bahmani, Benjamin Moseley, Andrea Vattani, Ravi Kumar, and Sergei Vassilvitskii. Scalable k-means++. *PVLDB*, 5(7):622–633, 2012.

[9] Moses Charikar, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. Incremental clustering and dynamic information retrieval. In *Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing*, STOC '97, pages 626–635, New York, NY, USA, 1997. ACM.

[10] Anna Choromanska and Claire Monteleoni. Online clustering with experts. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2012, La Palma, Canary Islands, April 21-23, 2012*, pages 227–235, 2012.

[11] Adam Coates, Andrew Y. Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In Geoffrey J. Gordon, David B. Dunson, and Miroslav Dudík, editors, *AISTATS*, volume 15 of *JMLR Proceedings*, pages 215–223. JMLR.org, 2011.

[12] Sanjoy Dasgupta. Topics in unsupervised learning. Class Notes CSE 291, 2014.

[13] Zvi Drezner and Horst W. Hamacher. *Facility location - applications and theory*. Springer, 2002.

[14] Rong-En Fan. Libsvm data: Classification, regression, and multi-label., 2014.

[15] Dimitris Fotakis. On the competitive ratio for online facility location. *Algorithmica*, 50(1):1–57, 2008.

[16] Dimitris Fotakis. Online and incremental algorithms for facility location. *SIGACT News*, 42(1):97–131, 2011.

[17] Sudipto Guha, Adam Meyerson, Nina Mishra, Rajeev Motwani, and Liadan O'Callaghan. Clustering data streams: Theory and practice. *IEEE Trans. Knowl. Data Eng.*, 15(3):515–528, 2003.

[18] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. In *Symposium on Computational Geometry*, pages 10–18, 2002.

[19] Percy Liang and Dan Klein. Online EM for unsupervised models. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, May 31 - June 5, 2009, Boulder, Colorado, USA*, pages 611–619, 2009.

[20] Stuart P. Lloyd. Least squares quantization in pcm. *IEEE Trans. Inf. Theor.*, 28(2):129–137, September 1982.

[21] Adam Meyerson. Online facility location. In *FOCS*, pages 426–431. IEEE Computer Society, 2001.

[22] Adam Meyerson, Michael Shindler, and Alex Wong. Fast and accurate k-means for large datasets. *NIPS*, 2011.

[23] Rafail Ostrovsky, Yuval Rabani, Leonard J. Schulman, and Chaitanya Swamy. The effectiveness of lloyd-type methods for the k-means problem. *J. ACM*, 59(6):28, 2012.

[24] Jens Vygen. Approximation algorithms for facility location problems. Lecture Notes, Technical Report No. 05950, 2005.