$\nabla_b f(w,b) = \frac{\partial f(w,b)}{\partial b} = C\sum_{i=1}^{n} \frac{\partial L(x_i,y_i)}{\partial b}$, where $\frac{\partial L(x_i,y_i)}{\partial b} = \begin{cases} 0 if y_i(x_i * w + b) \geq 1 \\ -y_i otherwise \end{cases}$.

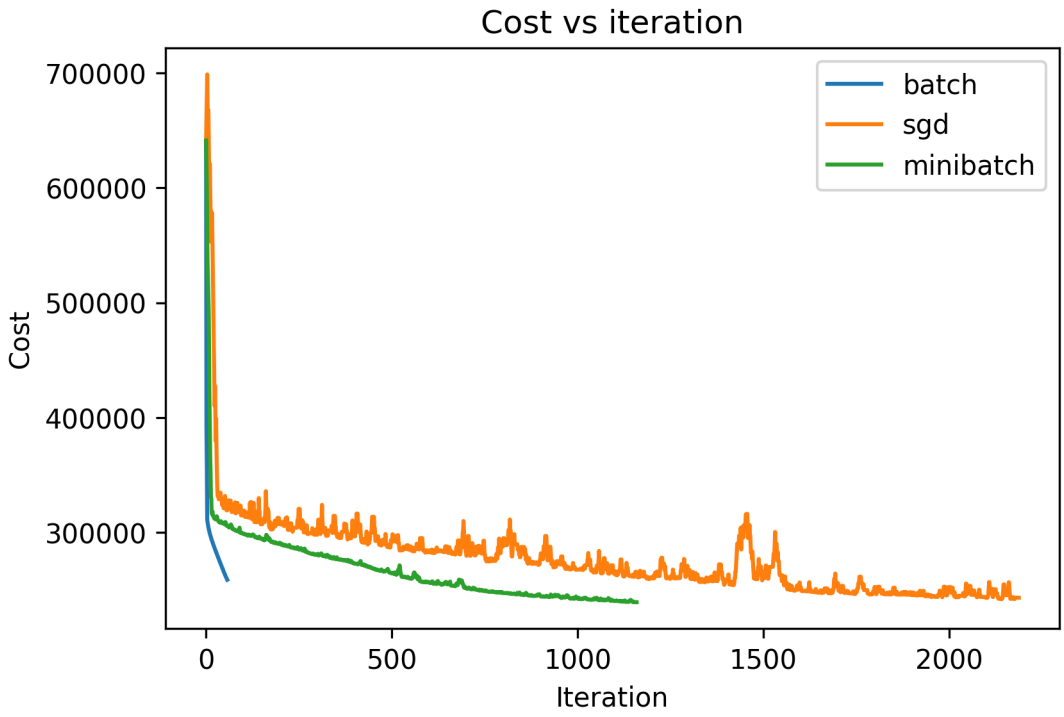

Figure 1: Cost vs Iterations

Run name	Time (s)	# iterations	# epochs
Batch	0.37	57	57
SGD	4.38	2188	0.34
Mini Batch	2.44	1159	3.61

Table 1: Batch, SGD and Mini batch

When the batch size increases , we see the following effects:

- The training curve becomes smoother.
- The training time decreases.
- The number of iterataions decreases,
- The dataset is traversed for more rounds.

The batch GD was the fastest, because the data was vectorized and could fit in memory, loss was calculated for the fewest times per data point. If these factors could be leveled, we should see SGD being the fastest, which corresponds to the fewest number of epochs required.

$I(D) = 100 * (1 - 0.4^2 - 0.6^2) = 48$

Wine: $48 - 50 * (1 - \frac{2}{5}^2 - \frac{3}{5}^2) - 50 * (1 - \frac{2}{5}^2 - \frac{3}{5}^2) = 0$

Running: $48 - 30 * (1 - \frac{2}{3}^2 - \frac{1}{3}^2) - 70 * (1 - \frac{2}{7}^2 - \frac{5}{7}^2) = \frac{128}{21} \approx 6.1$

Pizza: $48 - 80 * (1 - \frac{3}{8}^2 - \frac{5}{8}^2) - 20 * (1 - \frac{1}{2}^2 - \frac{1}{2}^2) = \frac{1}{2} = 0.5$

We should use the Running attribute.

a_1 will be the root. Left branch of the ($a_1 = 0$), around 99% of the leaves will be negative, on the right branch same amount around 99% will be positive. Therefore gain G is maximized. The two subtrees will have the same height, because we use all attributes, but not all branches have the same number of levels, because we stop branching when the population is too small to avoid overfitting. The desired decision tree which avoids overfitting would have had a single decision at the root which corresponds to a_i where we can assume 1% is the noise.

The first term on the RHS:

$$2cost_w(\widehat{S}, T) = 2 \sum_{t_{i,j \varepsilon \widehat{S}}} |S_{i,j}| \, d^2(t_{ij}, T) = 2 \sum_{x \varepsilon S} d^2(t_{ij}, T)$$

Where the last equation reads: for each x in S, x corresponds to an S_i, which corresponds to a set of t_{ij}. We sum up the squared distances from t_{ij} to T.

The second term of the RHS:

$$2 \sum_{i=1}^{l} cost(S_i, T_i) = 2 \sum_{i=1}^{l} \sum_{x \varepsilon S_i} d^2(x, T_i) = 2 \sum_{x \varepsilon S} d^2(x, t_{ij})$$

where the last equation is: for each x in S, x corresponds to an S_i, which corresponds to a set of t_{ij}. We sum up the squared distance from x to t_{ij}.

The suggested inequality gives:

$$2d^2(t_{ij}, T) + 2d^2(x, t_{ij} \geq (d(t_{ij}, T) + d(x, t_{ij}))^2$$

Because $d(x, t_{ij})$ is the distance from x to its associated t_{ij}, the triangular inequality gives us

$$d(t_{ij}, T) + d(x, t_{ij}) \geq d(x, T)$$

Stringing all the above gives the desired inequality.

$\sum_{i=1}^{l} cost(S_i, T_i)$

$= \sum_{i=1}^{l} cost(S_i, T_i^*)\alpha$ (T_i is found by ALG, T_i^* means the optimal for S_i)

$\leq \sum_{i=1}^{l} cost(S_i, T_i)\alpha$ (T_i^* is the optimal)

$\leq \alpha \times cost(S, T^*)$ (Group all x in each S_i)

Summing up over all i gives the result.

First inequality:

$cost_w(\widehat{S}, T)$

$\leq \alpha \times cost_w(\widehat{S}, \widehat{T}^*)$ (T is found by ALT. $\widehat{T}$ means the optimal for the $\widehat{S}$.)

$\leq \alpha \times cost_w(\widehat{S}, T^*)$ ($\widehat{T}^*$ is optimal for $\widehat{S}$, but T is not)

Second inequality follows similalrly from (a), by replacing T with T^*. Similarly to part (a), where $\forall x \varepsilon S_{ij} (1 \leq i \leq l, 1 \leq j \leq k)$ we have some: $d(t_{ij}, T^*)^2 \leq 2d(t_{ij}, x)^2 + 2d(x, T^*)^2$. Therefore by (a) and (b) and the above:

$$
\begin{aligned}
& cost(S, T) \\
& \leq 2 \times cost_w(\widehat{S}, T) + 2 \sum_{i=1}^{l} cost(S_i, T_i) \\
& \leq 4\alpha(\sum_{i=1}^{l} cost(S_i, T_i + cost(S, T^*)) + 2 \sum_{i=1}^{l} cost(S_i, T_i) \\
& \leq (4\alpha(\alpha + 1) + 2\alpha) cost(S, T^*) \\
& \leq (4\alpha^2 + 6\alpha) cost(S, T^*)
\end{aligned}
$$

$P_r[\tilde{F}[i] \leq F[i] + \epsilon t]$

$= P_r[min_j C_{j,h_j(i)} \leq F[i] + \epsilon t]$ (by definition)

$= P_r[\exists j \ni C_{j,h_j(i)} \leq F[i] + \epsilon t]$ (set logic)

$= 1 - P_r[\forall j, C_{j,h_j(i)} > F[i] + \epsilon t]$ (complement)

$= 1 - \prod_{j=1}^{[log\frac{1}{\partial}]} P_r[C_{j,h_j(i)} > F[i] + \epsilon t]$ (independence of hash funcitions)

We now find $P[\cdot]$

$P_r[C_{j,h_j(i)} > F[i] + \epsilon t]$

$\leq P_r[C_{j,h_j(i)} \geq F[i] + \epsilon t]$

$= P_r[C_{j,h_j(i)} - F[i] \geq \epsilon t]$

$\leq \frac{E[C_{j,h_j(i)} - F[i]]}{\epsilon t}$ (Markov's inequality. Random variable > 0 by property 1.)

$\leq \frac{t - F[i]}{\epsilon t}$ (property 2)

$\leq \frac{1}{\epsilon}$ $(F[i] \geq 0)$

Back to the main derivation:

$$P_r[\tilde{F}i \leq F[i] + \epsilon t] > 1 - (\frac{1}{epsilon})^{log\frac{1}{\partial}} = 1 - \partial$$

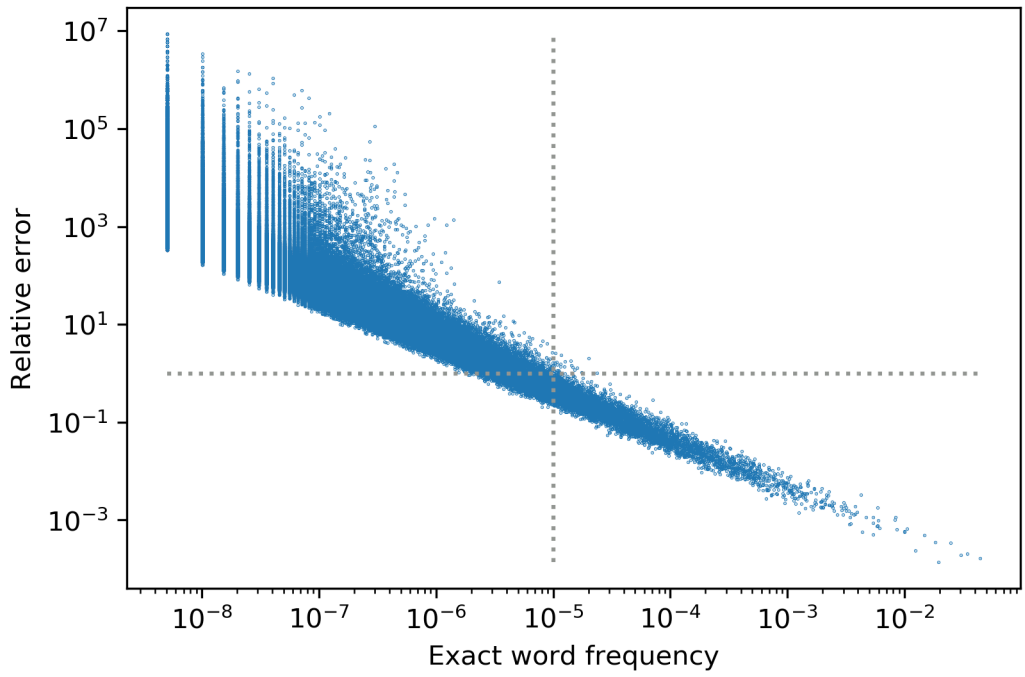

Figure 2: Results

The relative error tends to be < 1, if the exact frequency is $> 1e^{-5}$. This is consistent with the parameters. The algorithm also works best for frequent words, which can be seen in figure 2.

Information sheet
CS246: Mining Massive Data Sets

Assignment Submission Fill in and include this information sheet with each of your assignments. This page should be the last page of your submission. Assignments are due at 11:59pm and are always due on a Thursday. All students (SCPD and non-SCPD) must submit their homework via Gradescope (http://www.gradescope.com). Students can typeset or scan their homework. Make sure that you answer each (sub-)question on a separate page. That is, one answer per page regardless of the answer length. Students also need to upload their code on Gradescope. Put all the code for a single question into a single file and upload it.

Late Homework Policy Each student will have a total of *two* late periods. *Homework are due on Thursdays at 11:59pm PT and one late period expires on the following Monday at 11:59pm PT*. Only one late period may be used for an assignment. Any homework received after 11:59pm PT on the Monday following the homework due date will receive no credit. Once these late periods are exhausted, any assignments turned in late will receive no credit.

Honor Code We strongly encourage students to form study groups. Students may discuss and work on homework problems in groups. However, each student must write down their solutions independently, i.e., each student must understand the solution well enough in order to reconstruct it by him/herself. Students should clearly mention the names of all the other students who were part of their discussion group. Using code or solutions obtained from the web (GitHub/Google/previous year's solutions etc.) is considered an honor code violation. We check all the submissions for plagiarism. We take the honor code very seriously and expect students to do the same.

Your name: _____

Email: _____ **SUID:** _____

Discussion Group: _____

I acknowledge and accept the Honor Code.

(Signed) _____