Homework Assignment 2

Anurag, arp3969

Please upload the HW on canvas by 10pm Nov 14th. Please type up your homework using latex. We will not accept handwritten homeworks[1].

1. Consider the following problem of fused Lasso, or total variation de-noising.

$$\min_x \frac{1}{2}\|x - z\|_2^2 + \lambda \sum_{i=1}^{n-1} |x_{i+1} - x_i|.$$

Here, z is a noisy signal and λ is non-negative.

(a) Write this problem as follows:

$$\min_x \frac{1}{2}\|x - z\|_2^2 + \lambda\|Dx\|_1.$$

where D is a $n - 1 \times n$ matrix. What is D?

$$D = \begin{bmatrix} -1 & 1 & 0 & ... & 0 \\ 0 & -1 & 1 & ... & 0 \\ 0 & 0 & -1 & ... & 0 \\ 0 & 0 & .. & -1 & 1 \end{bmatrix}$$

(b) Write down the subgradient of the objective function.

$$\text{g} = x - z + \lambda D^T \gamma$$

where γ is $sign(Dx_i)$ and [-1,1] if Dx$_i = 0$

(c) Implement the subgradient descent algorithm. On the noisy.txt dataset, apply the algorithm and show the convergence plot against number of iterations. Play with different stepsizes and discuss how that affects the convergence.

Ans:

[1] Two of these homeworks were adapted from A. Dimakis and C. Caramanis's class

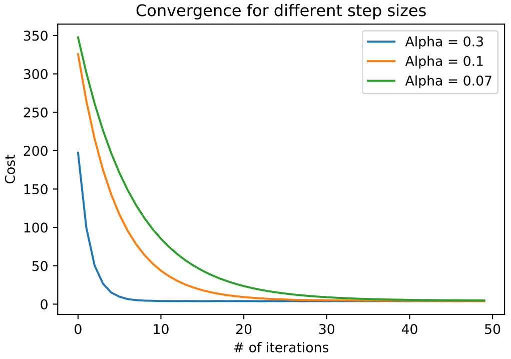

Figure 1: cost as a function of iterations

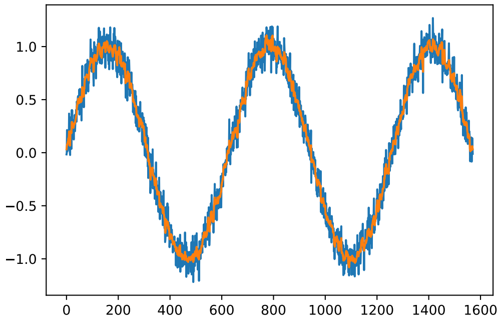

Figure 2: denoised data superimposed

(d) The problem can be written as

$$\min_x \frac{1}{2}\|x - z\|_2^2 + \lambda\|y\|_1$$
$$\text{s.t. } y = Dx \tag{1}$$

Show that the dual of the above problem is:

$$\min_u \frac{1}{2}u^T DD^T u - u^T Dz.$$
$$\text{s.t. } \|u\|_\infty \leq \lambda \tag{2}$$

Ans: The Lagrangian of the above problem can be written as:

$$\text{L}(x, u^T) = \frac{1}{2}\|x - z\|_2^2 + \lambda\|y\|_1 + u^T(Dx - y)$$

Thus the problem is to,

$$\min_x \max_{u^T} \frac{1}{2}\|x - z\|_2^2 + \lambda\|y\|_1 + u^T(Dx - y)$$

Thus, differentiating w.r.t x and setting it to 0. we get x* as:

$$\mathrm{x}^* = z - \lambda D^T\gamma - D^T u$$

Plugging this in the equation we get,

$$= \min_{u^T} \frac{1}{2}\|z - \lambda D^T\gamma - D^T u - z\|_2^2 + \lambda\|y\|_1 + u^T(D(z - \lambda D^T\gamma - D^T u) - y)$$
$$= \min_{u^T} \frac{1}{2}(\lambda D^T\gamma + D^T u)^T(\lambda D^T\gamma + D^T u) + \lambda\|y\|_1 - u^T y + u^T Dz - \lambda u^T DD^T\gamma - u^T DD^T u$$
$$= \min_{u^T} \frac{1}{2}(u^T DD^T u)^T + u^T Dz + (\lambda\|y\|_1 - u^T y) \tag{3}$$

The last term can be written as $\|u\|_\infty < \lambda$ as it would take the value of ∞ if the condition is violated and value otherwise

(e) Now implement the proximal gradient algorithm for the dual problem. In order to do this, you may need to look at the proximal operator given below:

$$\mathrm{prox}_t(x) = \arg\min_z \frac{\|x - z\|^2}{2t} + I_\lambda(z)$$

where $I_\lambda(z) = 0$ if $|z| \le \lambda$ and ∞ otherwise. In this case, the proximal operator gets reduced to a projection operator. On the same datset, apply this algorithm and show the convergence plot against the number of iterations. How does this compare with the subgradient method you implemented before?
Ans:

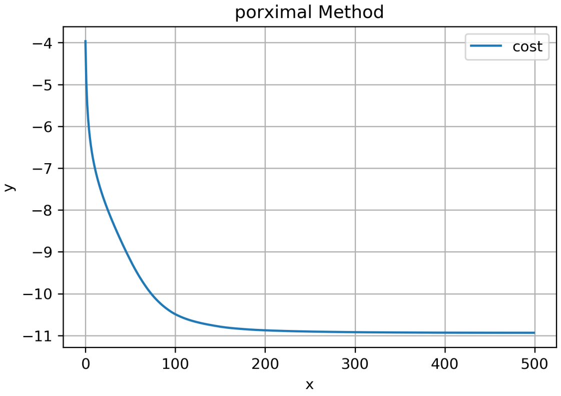

Figure 3: cost as a function of iterations

As can be seen from the plots, the dual proximal gradient descent algorithm is faster. The upside of the subgradient method is that the gradient g is easy to compute, provided the gradient of the function is easy to compute. Also,additional advantage of subgradient is that the iterations are cheap (we sum up rows of D over active set S).But the drawback is slower convergence. For the dual proximal gradient, the iterations involve projecting onto a box, hence, are cheap as well with medium convergence

(f) Finally, for the dual proximal gradient method, show the effect of different λ values by plotting the denoised curve superimposed on the original noisy curve.
Ans:

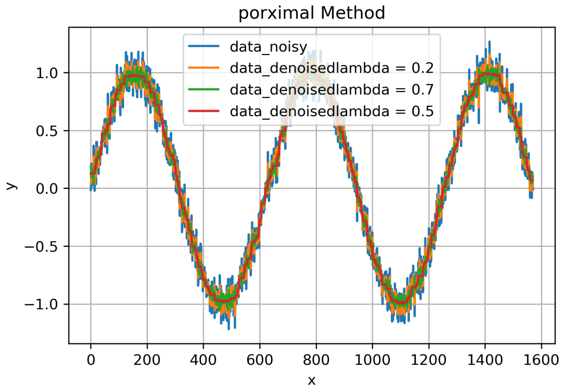

Figure 4: denoised data superimposed

It can be seen that as the value of λ increase the the denoising increases up until a certain point Beyond this point the denoising starts to decrease again. Hence, there is a range of optimal values of λ

2. Consider the following two similarity functions for two sets A and B. The overlap similarity function:

$$sim_{over}(A,B) = \frac{|A \cap B|}{\min(|A|,|B|)}$$

and the Dice similarity function

$$sim_{dice}(A,B) = \frac{2|A \cap B|}{|A|+|B|}$$

Is there a locality sensitive scheme for these? Prove or give a counter example.
Ans: Let us prove that for a given collection of objects S,
A valid locality sensitive hashing scheme consist of hash functions for which 1-sim, the distance metric, satisfies the triangle inequality.

$$1 - sim(x,y) + 1 - sim(y,z) \geq 1 - sim(x,z)$$

Consider the sets X= (1), Y = (2) and Z=(1,2)
then for these,

$$sim_{dice}(X,Z) = \frac{2}{3} \tag{4}$$

$$sim_{dice}(Z,Y) = \frac{2}{3} \tag{5}$$

$$sim_{dice}(X,Y) = 0 \tag{6}$$

Therefore, the triangle inequality does not hold true for the dice similarity function.

$$1 - sim(X,Z) + 1 - sim(Z,Y) \leq 1 - sim(X,Y)$$

For the overlap function,

$$sim_{over}(X,Z) = 1 \tag{7}$$

$$sim_{over}(Z,Y) = 1 \tag{8}$$

$$sim_{over}(X,Y) = 0 \tag{9}$$

Therefore, the triangle inequality does not hold true for the overlap similarity function.

$$1 - sim(X, Z) + 1 - sim(Z, Y) \leq 1 - sim(X, Y)$$

3. Exercises 3.6.3 and 3.6.4 from Chapter 3 of the book “Mining of Massive Datasets.”
Ans 3.6.3:Twice Differentiating the equation: $f(p) = 1 - (1 - p^4)^4$
we get,

$$f'(p) = -4(1 - p^4)^3(-4p^3) = 0$$
$$f''(p) = 16[-12(1 - p^4)^{p6} + (1 - p^4)^3 3p^2] = 0$$

The roots of this equation are p equals to $\frac{1}{5^{\frac{1}{4}}}$.
The value of the slope at this point is 1.19
Twice Differentiating the equation: $f(p) = (1 - (1 - p)^4)^4$
i.e $\frac{df(p)}{dp} = 0$ we get,

$$f'(p) = 16(1 - (1 - p)^4)^3(1 - p)^3 = 0$$
$$f''(p) = 16[12(1 - (1 - p)^4)^2(1 - p)^6 - 3(1 - p)^2(1 - (1 - p)^4)^3] = 0$$

The roots of this equation are p equals to 1-$\frac{1}{5^{\frac{1}{4}}}$.
The value of the slope at this point is: 2.44

Ans 3.6.4:

For the case of an r-way AND construction followed by a b-way OR construction, twice Differentiating the equation: $f(p) = 1 - (1 - p^r)^b$

$$f'(p) = -b(1 - p^r)^{b-1} - rp^{r-1} = 0$$
$$f''(p) = -b(b - 1)(1 - p^r)^{b-2}(rp^{r-1})^2 + (b(1 - p^r)^{b-1}r(r - 1)p^{r-2} = 0$$
$$\frac{(b - 1)r}{r - 1} = \frac{1 - p^r}{p^r}$$
$$\mathrm{p}^r = \frac{r - 1}{(b - 1)r + r - 1}$$

Substituting the value give the value of slope as
$-b(1 - \frac{r-1}{(b-1)r+r-1})^{b-1} - r\frac{r-1}{(b-1)r+r-1}^{r-1/r}$
For the case of a b-way OR construction followed by an r-way AND construction, twice Differentiating the equation: $f(p) = (1 - (1 - p)^b)^r$

$$f'(p) = r(1 - p^b)^{r-1}b(1 - p)^{b-1} = 0$$
$$f''(p) = r(r - 1)(1 - (1 - p)^b)^{r-2}(b(1 - p)^{b-1})^2 = 0$$
$$\frac{(1 - (1 - p)^b)}{(1 - p)^b} = \frac{(r - 1)b}{b - 1}$$
$$(1 - p)^b = \frac{b - 1}{(r - 1)b + b - 1}$$

Substituting the value give the value of slope as
$r(\frac{b-1}{(r-1)b+b-1})^{r-1}b(\frac{b-1}{(r-1)b+b-1})^{b-1/b}$

4. Download the dataset articles-1000.txt (1.6MB), which contains 1000 articles. Each row corresponds to an article, represented by an ID (e.g., t120) and its content (e.g., The Supreme Court in Johnnesberg on Friday...).

   (a) Convert each article into a set of 2-shingles (bigrams). The goal is to map each article ID to a set[2] of IDs of shingles that article contains. You can do this by going over the data once as follows: for each article, first split[3] it into a list of words, remove the stopwords[4] , generate a list of 2-shingles, and then hash each shingle to a 32-bit integer (i.e., the shingle ID) using CRC32 hash. The resulting shingle IDs range from 0 to $2^{32}-1$. What is the total number of unique shingles across all the books? What is the average number of shingles present per book?

   Here is an example of a 2-shingle and its hashed value (i.e., the shingle ID):

   ```
   import binascii
   shingle = "machine learning"
   shingleID = binascii.crc32(shingle) & 0xffffffff
   ```

   Ans: The total number of unique shingles are 145304. The average number of shingles present per book 254.8.

   (b) Generate MinHash signatures using 10 hash functions, where each is as follows: $h(x) = c_1x + c_2 \bmod p$.

   Set $p = 4294967311$, which is a prime number larger than $2^{32}-1$. Uniformly sample 10 values of $c_1, c_2 \in \{1, 2, ..., p-1\}$ and compute the corresponding MinHash signatures.

   For the 1st article, compare its signature vector with that of the rest of the articles and estimate their Jaccard similarity (i.e., compute the percentage of signatures that are equal). Find the book that has the largest (estimated) similarity with the 1st book then compute the actual Jaccard similarity between the two books (based on the computed shingle sets). Your result should be a triplet (bookID, estimatedJaccard, trueJaccard).

   Ans: The triplet is (t7998, 1, 0.9926470588235294). Please refer to the attached python notebook for the implementation.

   Hint: remember, you do not need to generate 10 permutations. Use the trick we learned in class, you can also find it in Section 3.3.5 of the "Mining of Massive Datasets" book linked from the class website.

   (c) **Amplification** Use the LSH technique in Section 3.4 of the same book, construct $n = br$ MinHash signatures, where b is the number of bands, and r is the number of hash values in each band. Find all the articles that are "similar" to the 1st article (i.e., articles that agree in at least one band of signatures), and put it into a set called S (excluding the 1st article itself). Let $Sim(i, j)$ be the actual Jaccard similarity between articles i and j. Given a threshold t, define the percentage of

[2] You can use Python sets to do that. A set should contain unique elements.

[3] Use Python String split() method to do that– no need to remove the punctuations.

[4] you can remove the stop words specified by stopwords.words("english") from nltk.corpus.

false positives (fp) in set S as

$$\text{False positives} = \frac{|\{i \in S : Sim(i,1) < t\}|}{|S|}.$$

Set t = 0.8, plot fp as a function of b (for $b = 1, 3, 5, 7, 9$ with $r = 2$). Similarly, plot fp as a function of r (for $r = 1, 3, 5, 7, 9$ with $b = 10$). For each (b, r) pair, plot the average value of fp over 10 realizations. Give a short comparison of the two.

Ans:

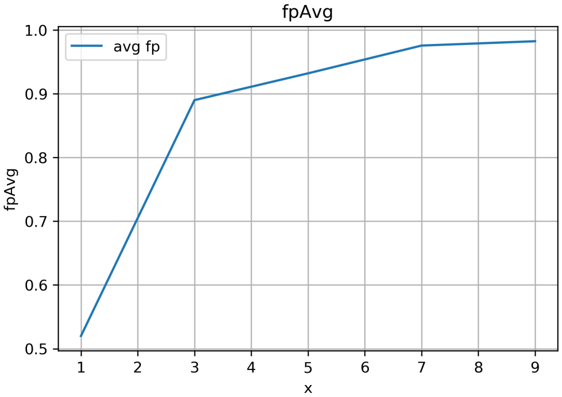

Figure 5: fp as a function of b

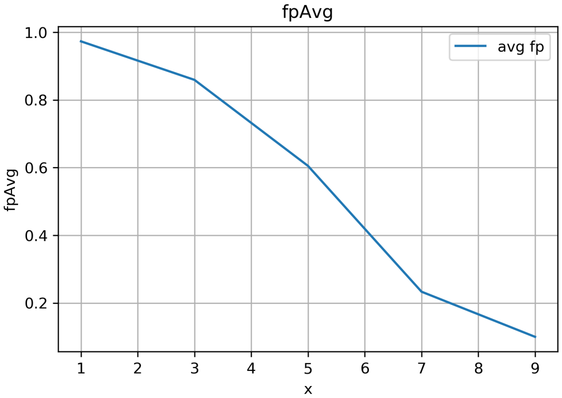

Figure 6: fp as a function of r

Any pair that hashed to the same bucket for any band is a candidate pair. It is expected that most of the dissimilar pairs will never hash to the same bucket and therefore will never be checked. Here, an analysis to verfiy this argument for different values of bands and rows and hence, different hash functions was made. For each band, there is a hash function that takes vectors of r integers and hashes them to some large number of buckets. The threshold is roughly where the rise is the steepest and for large b and r, there we find that the pairs with similarity above threshold are very likely to become candidate pairs. The threshold is $(1/b)^{1/r}$. Thus, here, for t = 0.8 and r =2 a good band value is approximately to 2 according to the theoretical result. When b = 10 and t = 0.8, a good r value comes close to 10. A similar trend is observed for the problem here.

(d) Find the twenty most similar article pairs without any approximation. How long did it take? Now find the twenty most similar article pairs using LSH (with b,r as hyperparameters). Compare your results for different (b, r) pairs.

Ans: For the twenty most similar article pairs without any approximation, it took 17.3 s 255 ms per loop (mean std. dev. of 7 runs, 1 loop each). For b = 3, r = 10, it took 1.13 s 101 ms per loop (mean std. dev. of 7 runs, 1 loop each)+time for calculating jaccard similarity . It is seen that as the b and r get away from the optimal theoritical values for the threshold, the iterations become faster if more rows are considered as there is an increased chance of finding a candidate pair, for which the article is removed in the next iteration. But this is at the cost of increased time spent on calculating the actual jaccard similarity later. This is still faster then computing jaccard value for all the pairs as lesser number of irrevelant pairs are considered for LSH.