Project 4: Report

Nanthini Balasubramanian and Harshat Kumar

I. Question 1.1

For the representation we used only the non-zero values i.e, if an user has rated the movie - he/she likes the movie. This way we generated a representation that allowed us to find the most popular movies.

II. Question 1.2

$F(A) = \frac{1}{n}\sum_{i=1}^{n} \max_{j\in A} r_{i,j}$ is submodular and monotone.

A. *Monotone*

Given sets $A \subseteq B$, we want to show $F(A) \leq F(B)$

proof:

Given user i, let us define $j^* = \arg\max_{j\in A} r_{i,j}$.

Then $r_{i,j^*} = \max_{j\in A} r_{i,j^*}$

So $F(A) = \frac{1}{n}\sum_{i=1}^{n} r_{i,j^*}$

Similarly, we can define $\hat{j} = \arg\max_{j\in B} r_{i,j}$.

Then $r_{i,\hat{j}} = \max_{j\in B} r_{i,\hat{j}}$

So $F(B) = \frac{1}{n}\sum_{i=1}^{n} r_{i,\hat{j}}$

We can partition the set B into two disjoint sets: $B = A \cup (B \cap A^c)$

This gives us two cases:

Case $\hat{j} \in A$: then $r_{i,j^*} = r_{i,\hat{j}}$

Case $\hat{j} \in B \cap A^c$: then $r_{i,j^*} \leq r_{i,\hat{j}}$ because if $r_{i,j^*} > r_{i,\hat{j}}$, then that would mean the maximum j was in A, which is a contradiction.

Hence we have shown for all i, $\max_{j\in A} r_{i,j} \leq \max_{j\in B} r_{i,j}$

And so

$$\frac{1}{n}\sum_{i=1}^{n} \max_{j\in A} r_{i,j} \leq \frac{1}{n}\sum_{i=1}^{n} \max_{j\in B} r_{i,j}$$
$$F(A) \leq F(B)$$

and we are done.

B. *Submodular*

Given sets $A \subseteq B$ and some $e \in \Omega$, we want to show $F(A \cup e) - F(A) \geq F(B \cup e) - F(B)$

proof:

Since $e \in \Omega$, there are 3 cases.

1) Case $e \in A$: Then $F(A \cup e) = F(A)$ and $F(B \cup e) = F(B)$. So the inequality

$$F(A \cup e) - F(A) \geq F(B \cup e) - F(B)$$

$$F(A) - F(A) \geq F(B) - F(B)$$

$$0 \geq 0$$

holds.

2) Case $e \in B \cap A^c$: Then $F(B \cup e) = F(B)$ and $A \subsetneq A \cup e$

Because F is monotone, $F(A \cup e) \geq F(A)$ so

$$F(A \cup e) - F(A) \geq 0 = F(B \cup e) - F(B)$$

and the inequality holds.

The final case is non-trivial.

3) Case $e \notin B$:

We will continue from our notation from before, namely:

$$j^* = \arg\max_{j \in A} r_{i,j}$$

and

$$\hat{j} = \arg\max_{j \in B} r_{i,j}$$

So we have

$$F(A \cup e) - F(A) \geq F(B \cup e) - F(B)$$

$$\frac{1}{n}\sum_{i=1}^{n} \max\{r_{i,e}, r_{i,j^*}\} - \frac{1}{n}\sum_{i=1}^{n} r_{i,j^*} \geq \frac{1}{n}\sum_{i=1}^{n} \max\{r_{i,e}, r_{i,\hat{j}}\} - \frac{1}{n}\sum_{i=1}^{n} r_{i,\hat{j}}$$

$$\frac{1}{n}\sum_{i=1}^{n} (\max\{r_{i,e}, r_{i,j^*}\} - r_{i,j^*}) \geq \frac{1}{n}\sum_{i=1}^{n} \left(\max\{r_{i,e}, r_{i,\hat{j}}\} - r_{i,\hat{j}}\right)$$

Therefore, we are left to show that for every i,

$$\max\{r_{i,e}, r_{i,j^*}\} - r_{i,j^*} \geq \max\{r_{i,e}, r_{i,\hat{j}}\} - r_{i,\hat{j}}$$

From the previous problem, we know that $r_{i,j^*} \leq r_{i,\hat{j}}$, therefore we are left with the following 3 cases.

i) Case $r_{i,e} \leq r_{i,j^*} \leq r_{i,\hat{j}}$:

$$\max\{r_{i,e}, r_{i,j^*}\} - r_{i,j^*} \geq \max\{r_{i,e}, r_{i,\hat{j}}\} - r_{i,\hat{j}}$$

$$r_{i,j^*} - r_{i,j^*} \geq r_{i,\hat{j}} - r_{i,\hat{j}}$$

$$0 \geq 0$$

So the inequality holds.

ii) Case $r_{i,j^*} \leq r_{i,e} \leq r_{i,\hat{j}}$:

$$\max\{r_{i,e}, r_{i,j^*}\} - r_{i,j^*} \geq \max\{r_{i,e}, r_{i,\hat{j}}\} - r_{i,\hat{j}}$$

$$r_{i,e} - r_{i,j^*} \geq r_{i,\hat{j}} - r_{i,\hat{j}}$$

$$r_{i,e} - r_{i,j^*} \geq 0$$

Which is true because of monotonicty, so the inequality holds.

iii) Case $r_{i,j^*} \leq r_{i,\hat{j}} \leq r_{i,e}$:

$$\max\{r_{i,e}, r_{i,j^*}\} - r_{i,j^*} \geq \max\{r_{i,e}, r_{i,\hat{j}}\} - r_{i,\hat{j}}$$

$$r_{i,e} - r_{i,j^*} \geq r_{i,e} - r_{i,\hat{j}}$$

subtract $r_{i,e}$ from both sides

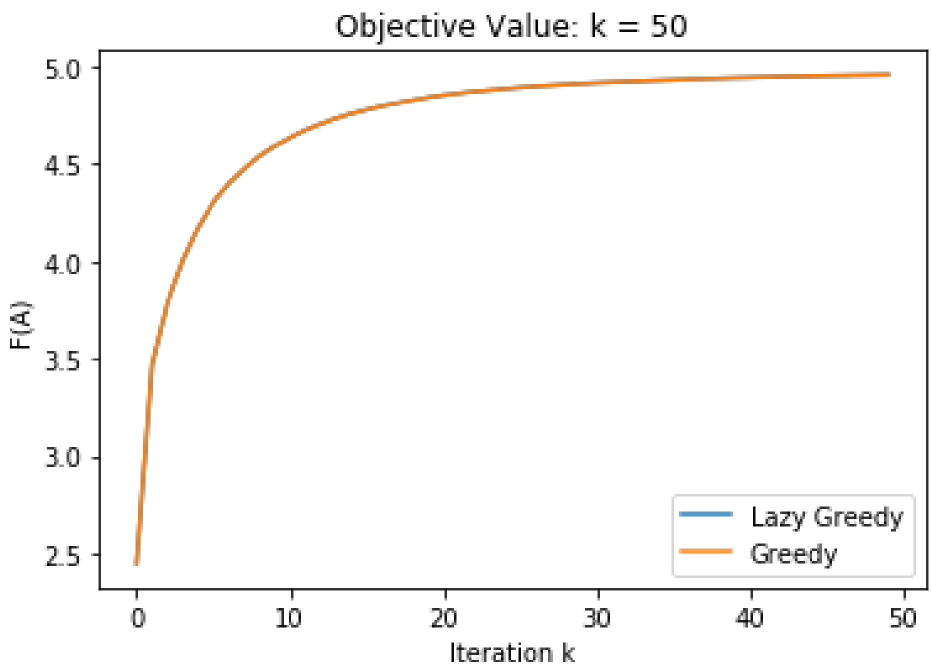
$$-r_{i,j^*} \geq -r_{i,\hat{j}}$$

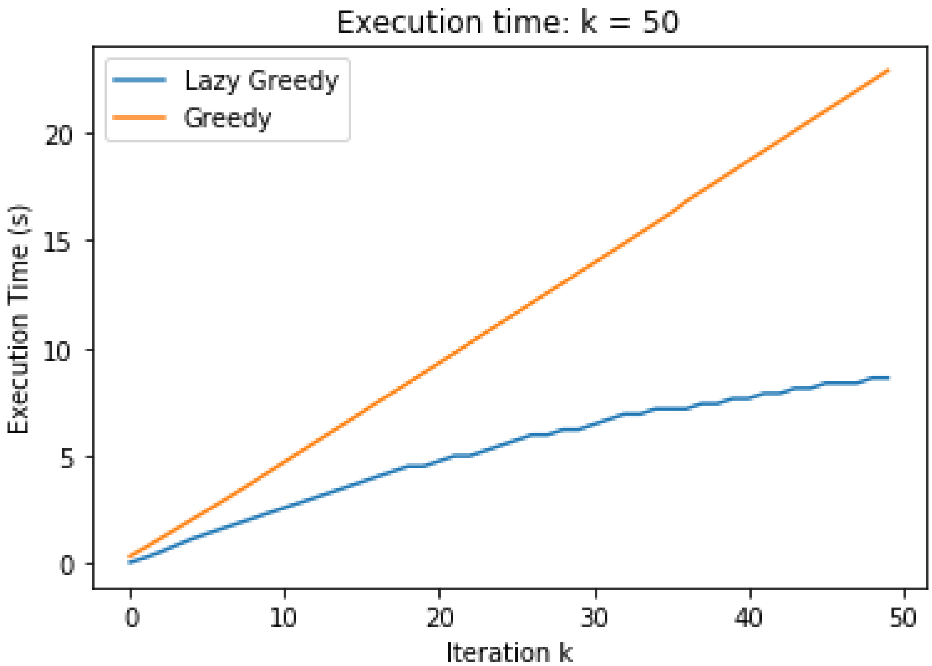
$$r_{i,\hat{j}} \geq r_{i,j^*}$$

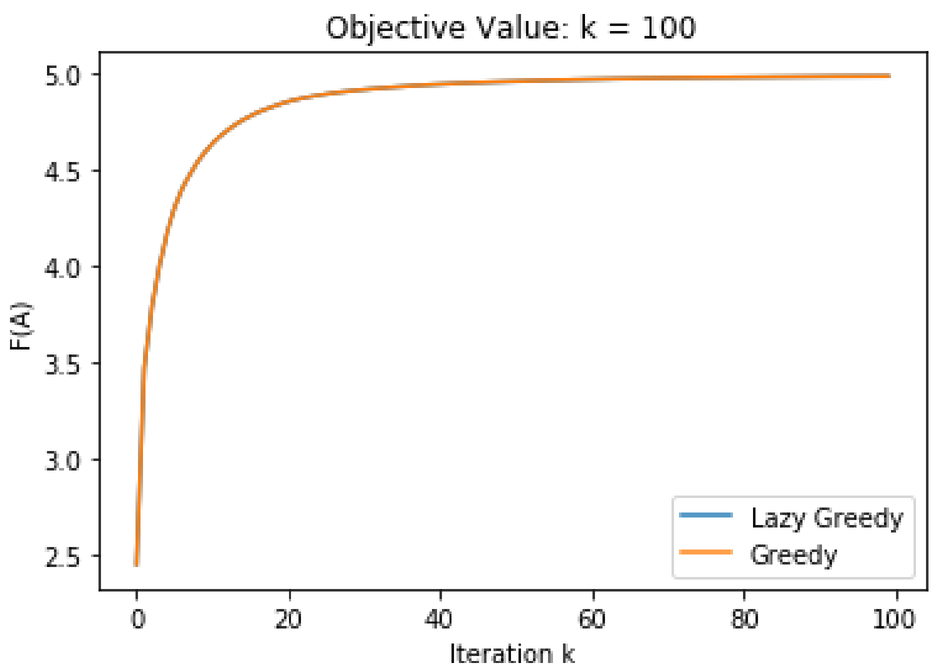
Which we showed in the previous section, so the inequality holds and we are done.

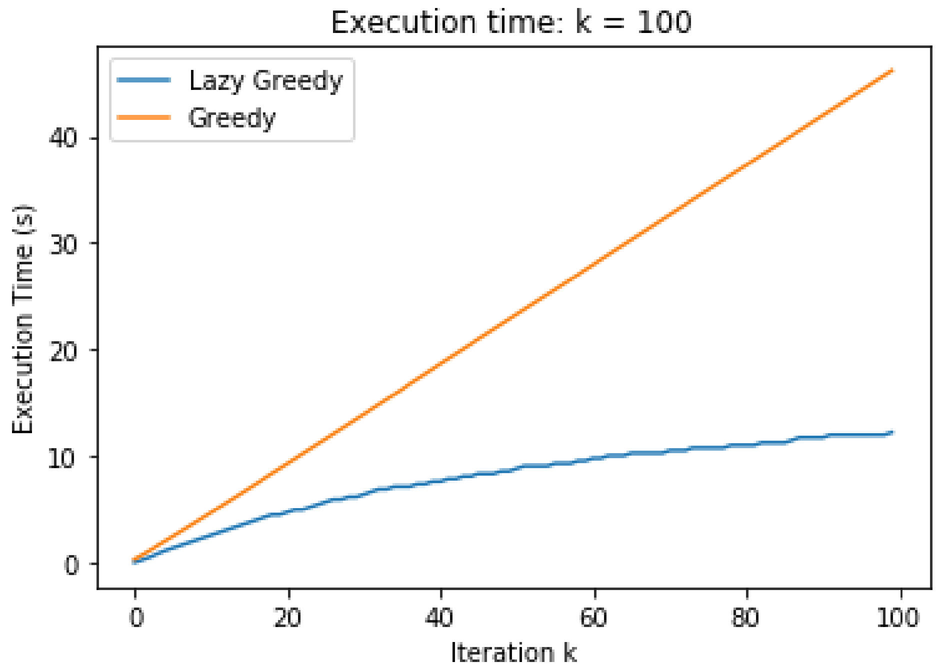
III. Question 1.3

The Greedy sub modular optimization and the 'Lazy' version of the algorithm was implemented and were compared in terms of convergence speed and performance. It is noted that the values obtained by them are the same - as expected to be but the convergence of Lazy Greedy Sub Modular Optimization is much faster than Greedy. This is because of the reduced number of comparisons being performed in Lazy Greedy.




Execution time: k = 50
Lazy Greedy
Greedy
Execution Time (s)
0
5
10
15
20
0
10
20
30
40
50
Iteration k


Objective Value: k = 100
F(A)
5.0
4.5
4.0
3.5
3.0
2.5
Lazy Greedy
Greedy
0
20
40
60
80
100
Iteration k


Execution time: k = 100
Lazy Greedy
Greedy
Execution Time (s)
Iteration k
0
10
20
30
40
0
20
40
60
80
100