Group name: Gravity Consulting
Group ID: 49

Introduction to Machine Learning(CS771)

Assignment 1

Rohit : 200813 : rohitg20@iitk.ac.in
Anshuman : 200157 : anshuman20@iitk.ac.in
Rishav : 200792 : rishav20@iitk.ac.in
Anoop : 200147 : anoopsr20@iitk.ac.in
Prakhar : 200696 : prakharg20@iitk.ac.in

---

1 First Exercise

As our feature map also contain either 0 or 1. so our ϕ function is such that it will return as same R vector.

Let **R** denotes the number of XOR gates contained by a single XORRO
Let $\mathbf{C} = \{C_0, C_1, C_2.....C_{r-1}\}$ denotes a vector of size R and contains either 0 or 1
And f is the frequency.

Here we try to find a linear model for a XORRO PUF which is made up of 2 XORROs having **R** XOR gates each.

let Upper XORRO = u
let Lower XORRO = l

let **response** be the output of the XORRO PUF, then

$$\text{response} = 0 \text{ when } f^u > f^l$$
$$\text{response} = 1 \text{ when } f^l < f^u$$

OR we can say that

$$\text{response} = 0 \text{ when } t^l > t^u$$
$$\text{response} = 1 \text{ when } t^u < t^l$$

So we expect our output (in form of signum function) as

$$response = \frac{1 + Sign(f^l - f^u)}{2}$$

OR

$$response = \frac{1 + Sign(t^u - t^l)}{2}$$

Let timetaken by the i th XOR gate of the upper XORRO is denoted by t_i^u

Let timetaken by the i th XOR gate of the lower XORRO is denoted by t_i^l

Let δ_{00}^i,δ_{01}^i,δ_{10}^i,δ_{11}^i, be the time that the i th XOR gate takes before giving its output when the input to that gate is, respectively 00, 01, 10 and 11.

For i th XOR of upper XORRO when $C_i = 1$

$$t_i^u = \delta_{11}^i + \delta_{01}^i = p_i$$

For i th XOR of upper XORRO when $C_i = 0$

$$t_i^u = \delta_{00}^i + \delta_{10}^i = q_i$$

So, For any i th XOR gate of upper XORRO we can write

$$t_i^u = C_i p_i + (1 - C_i) q_i$$

SIMILARLY

For i th XOR of lower XORRO when $C_i = 1$

$$t_i^l = \delta_{11}^i + \delta_{01}^i = s_i$$

For i th XOR of lower XORRO when $C_i = 0$

$$t_i^l = \delta_{00}^i + \delta_{10}^i = r_i$$

So, For any i th XOR gate of lower XORRO we can write

$$t_i^l = C_i s_i + (1 - C_i) r_i$$

let Δ_i be the difference between the time taken by the ith XOR of the upper XORRO and the i th XOR of the lower XORRO

$$\Delta_i = t_i^u - t_i^l$$

$$\Delta_i = C_i p_i + (1 - C_i) q_i - (C_i s_i + (1 - C_i) r_i)$$

$$\Delta_i = C_i (p_i - s_i) + (1 - C_i)(q_i - r_i)$$

$$\Delta_i = (q_i - r_i) + C_i (p_i - s_i - q_i + r_i)$$

Total time taken = Sum of Δ_i along each of the R XOR gates

$$\sum_{i=0}^{r-1} \Delta_i = \sum_{i=0}^{r-1} \left((q_i - r_i) + C_i \left(p_i - q_i - s_i + r_i \right) \right)$$

let $(p_i - q_i - s_i + r_i)$ denotes w_i

$$\sum_{i=0}^{r-1} \Delta_i = \sum_{i=0}^{r-1} (q_i - r_i) + C_0 w_0 + C_1 w_1 + C_2 w_2 + ... + C_{r-1} w_{r-1}$$

LET

$$\sum_{i=0}^{r-1} (q_i - r_i) = b$$

So

$$\sum_{i=0}^{r-1} \Delta_i = b + \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{r-1} \end{bmatrix} \begin{bmatrix} C_0 & C_1 & \cdots & C_{r-1} \end{bmatrix}$$

Therefore the above expression can be represented as

$$\sum_{i=0}^{r-1} \Delta_i = b + w^T C$$

Thus Output **response** can be expressed as

$$response = \frac{1 + Sign(\sum_{i=0}^{r-1} \Delta_i)}{2}$$

$$response = \frac{1 + Sign(b + w^T C)}{2}$$

2 Second Exercise

As Total number of XORRO used is 16.
Let C be the vector of size R and contains either 0 or 1

firstly we try to make a model for every pair(i,j) where i and j are the index of XORROs.so total no. of model required will be 256

i = decimal equivalent of (65,66,67,68)th bits of given 73 bit input data j = decimal equivalent of (69,70,71,72)th bits of given 73 bit input data let we take **i** th and **j** th XORRO and make a linear model for pair (i,j) it which gives the response OUT_{ij} when we give input C

So ,

$$OUT_{ij} = \frac{1 + Sign(t^i - t^j)}{2}$$

where t^m denotes the time taken by that XORRO

Now we take **j** th and **i** th XORRO and make another linear modelfor pair (i,j) and it gives the response OUT_{ji} when we give input C

So ,

$$OUT_{ji} = \frac{1 + Sign(t^j - t^i)}{2}$$

Now,

$$OUT_{ji} = \frac{1 - Sign(t^i - t^j)}{2}$$

$$OUT_{ji} = \frac{1 - Sign(t^i - t^j) - 1 + 1}{2}$$
$$OUT_{ji} = \frac{2 - (1 + Sign(t^i - t^j))}{2}$$
$$OUT_{ji} = 1 - \frac{(1 + Sign(t^i - t^j))}{2}$$
$$OUT_{ji} = 1 - OUT_{ij}$$

So we can say that we can skip the linear model of pair(j,i) and use the linear model of pair(i,j) to get the results of pair (j,i) just by flipping the bit of the pair(i,j) model (1 - OUT_{ij})

So Total number of pairs can be 16*16 and we know that i must be different than j.Thus number of pairs when i = j are 16. Now possible number of pairs can be 16*16 - 16 = 240

As we discussed above that pair(j,i) results can be obtained from the pair(i,j) model.So we don't make a model for pair(j,i) and use the model of pair(i,j). By doing so, we can reduce the number of model required from 256 to (240/2) which is 120. So we take an list of 120 models. For training each model we will divide the input list to 120 list. where each list will have its corresponding model data.

model number for pair(i,j) = i*(i-1)/2) + j
model number for pair(j,i) = i*(i-1)/2) + j

$$\begin{bmatrix} & & (j,i) & & \\ & & & & \\ & & & & \\ (i,j) & & & & \end{bmatrix}$$

For pair(j,i) we will flip the output bit (73rd bit) and store its data or row into the list corresponding to pair(i,j).So while training the output corresponding to pair(j,i) input will be checked against the flipped output bit of pair(j,i) input row.

Benefit of doing this is saving space and now each model is getting around double data to train.

3 Fourth Exercise

providing this data based on secret_train.dat and secret_test.dat file

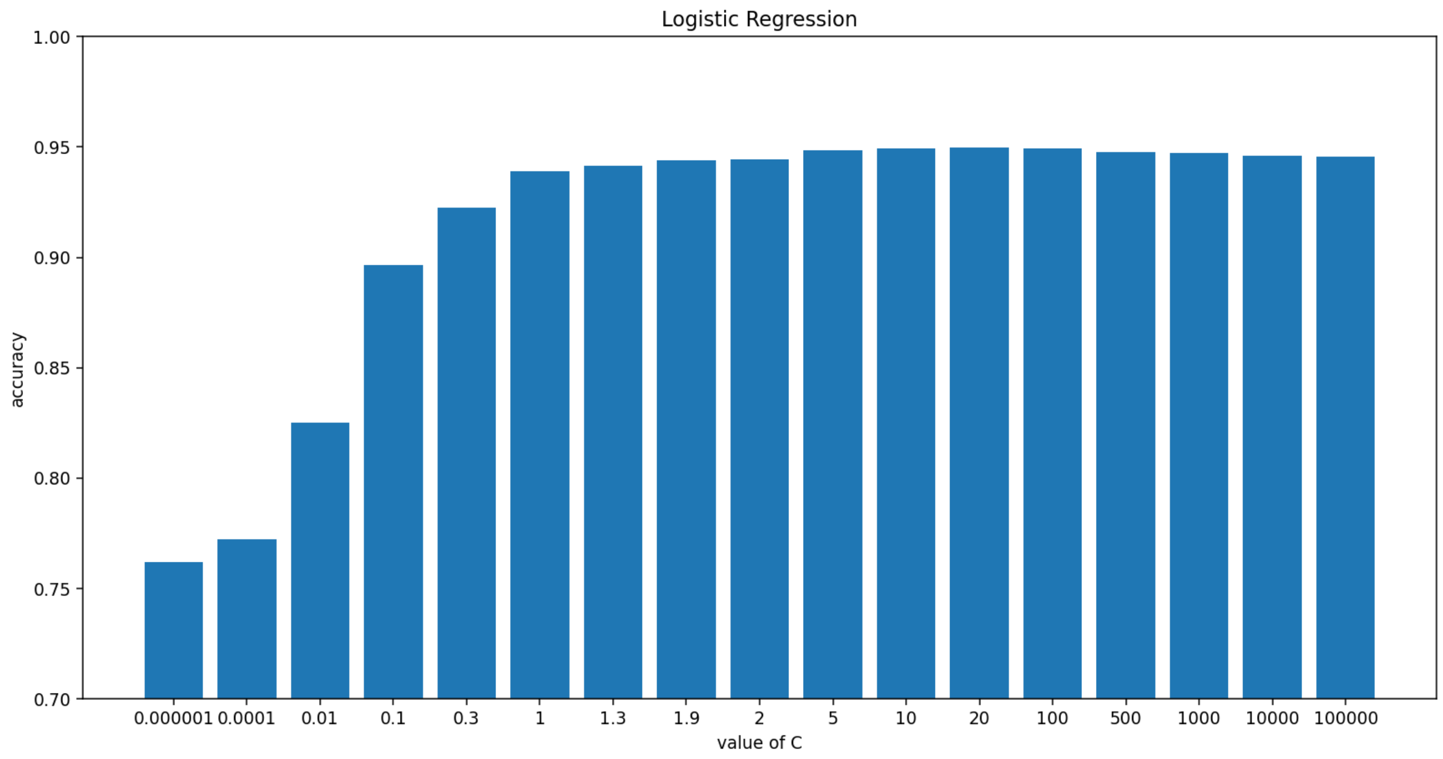
A) Loss hyperparameter in LinearSVC (hinge vs squared hinge)

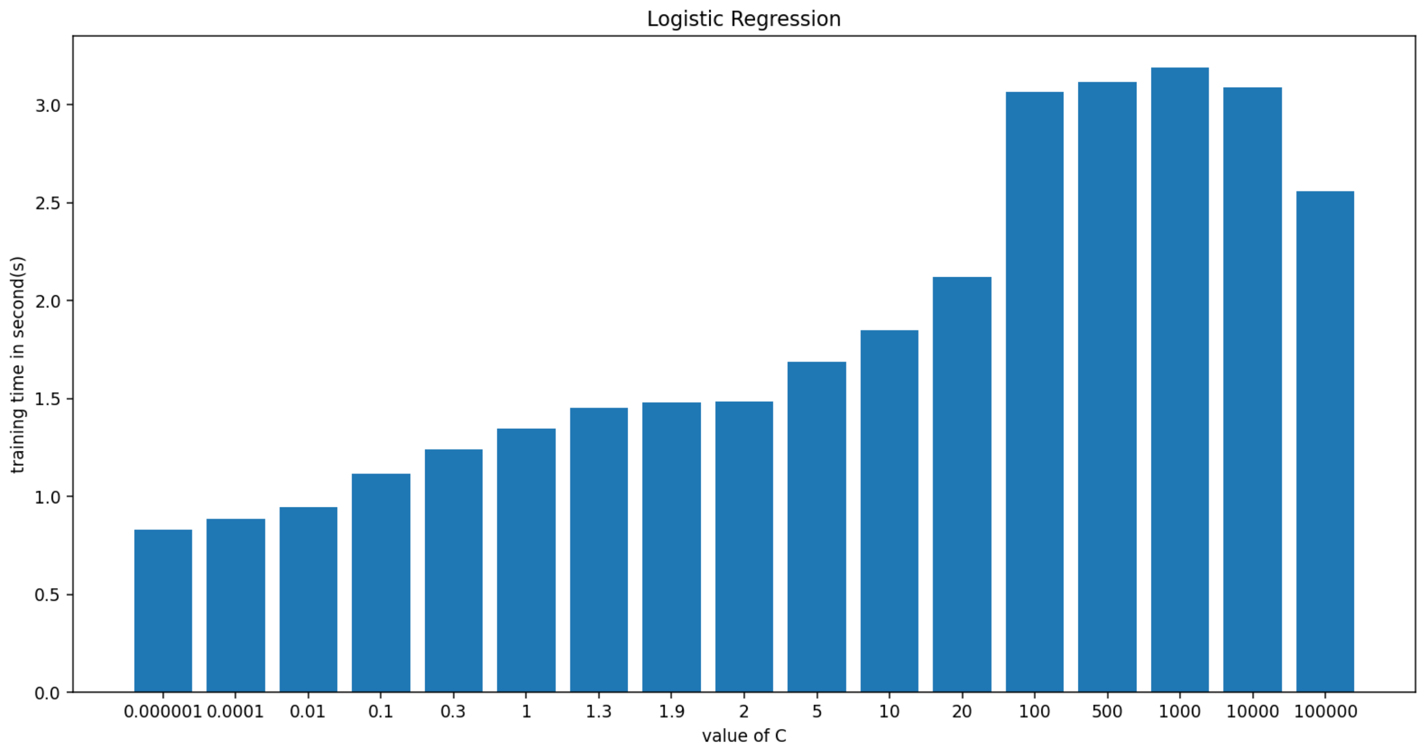
at values C = 0.3 and tol = 0.216

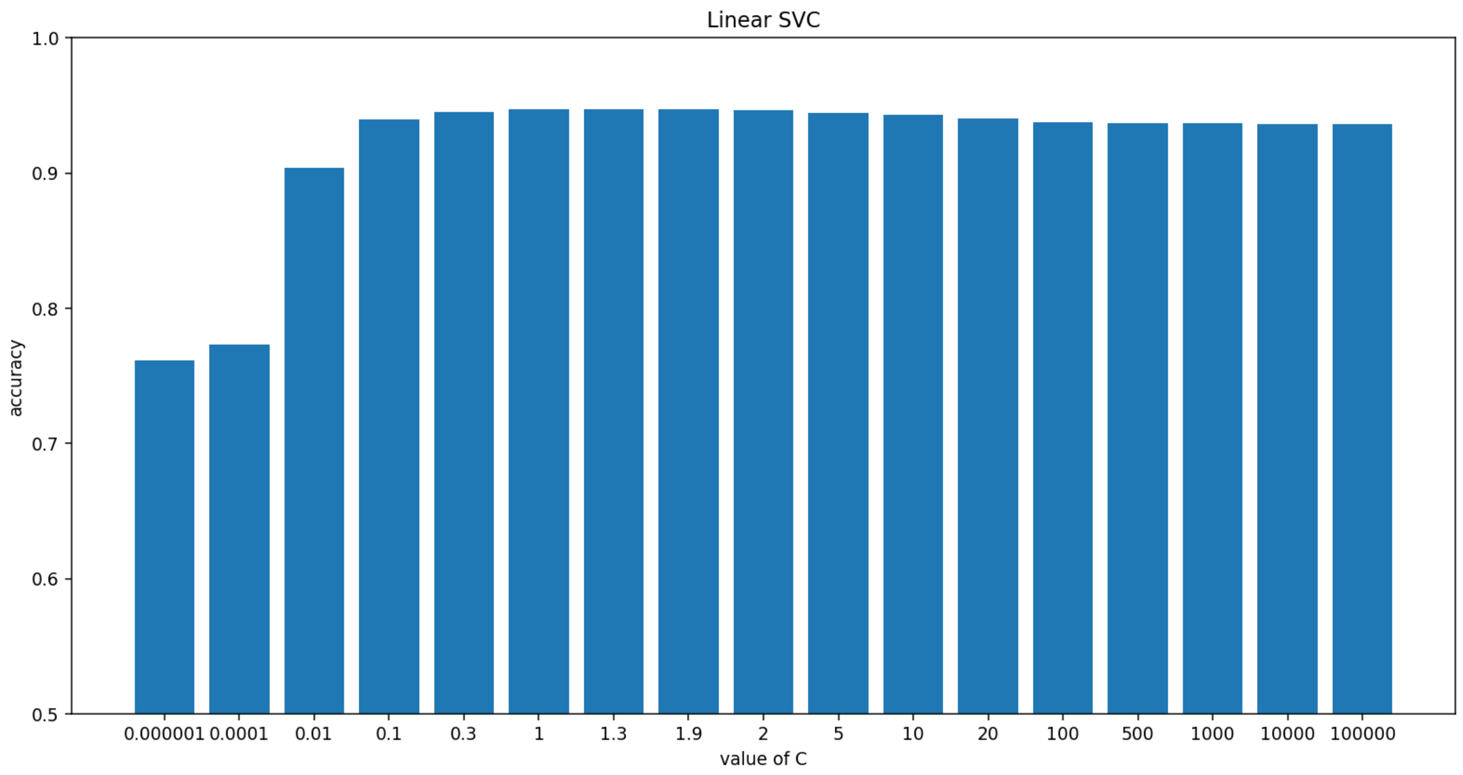
for hinge loss : training time is = 0.55887 and accuracy = 0.93445
for squared hinge loss : training time = 0.6157 and accuracy = 0.9454

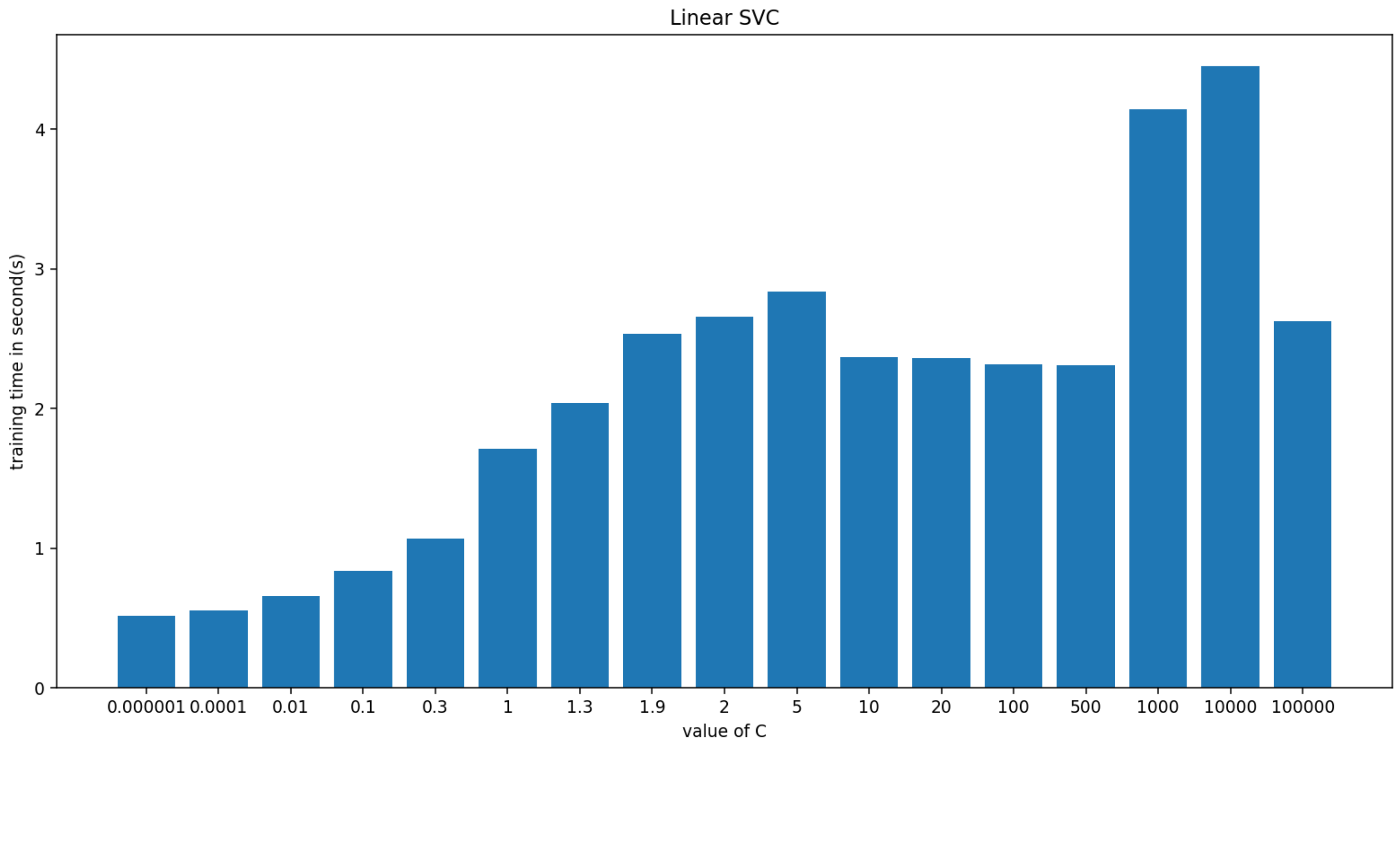
So, for hinge loss training time is less but accuracy is also less then squared hinge loss.

B) setting C hyperparameter in LinearSVC and LogisticRegression

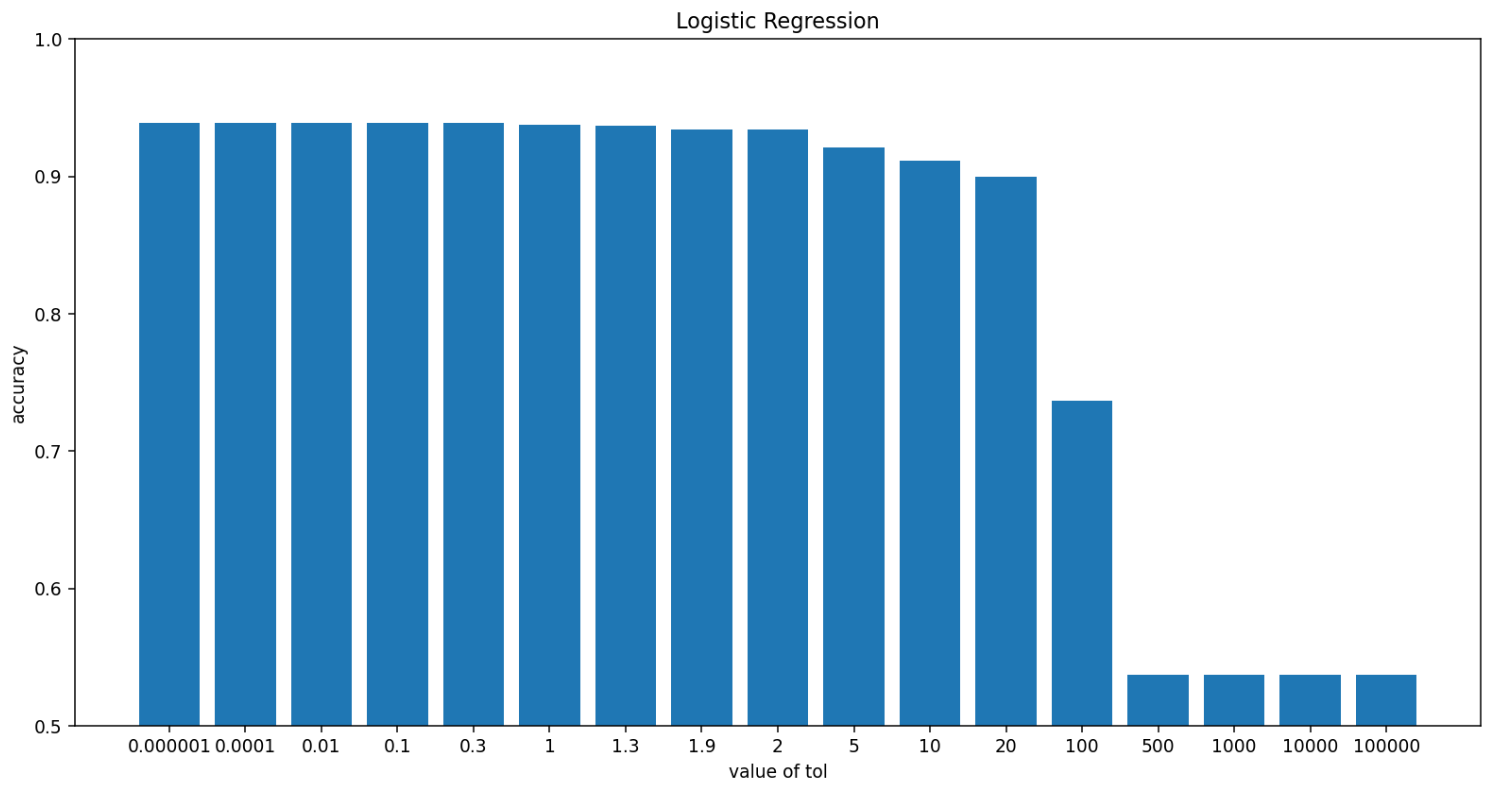


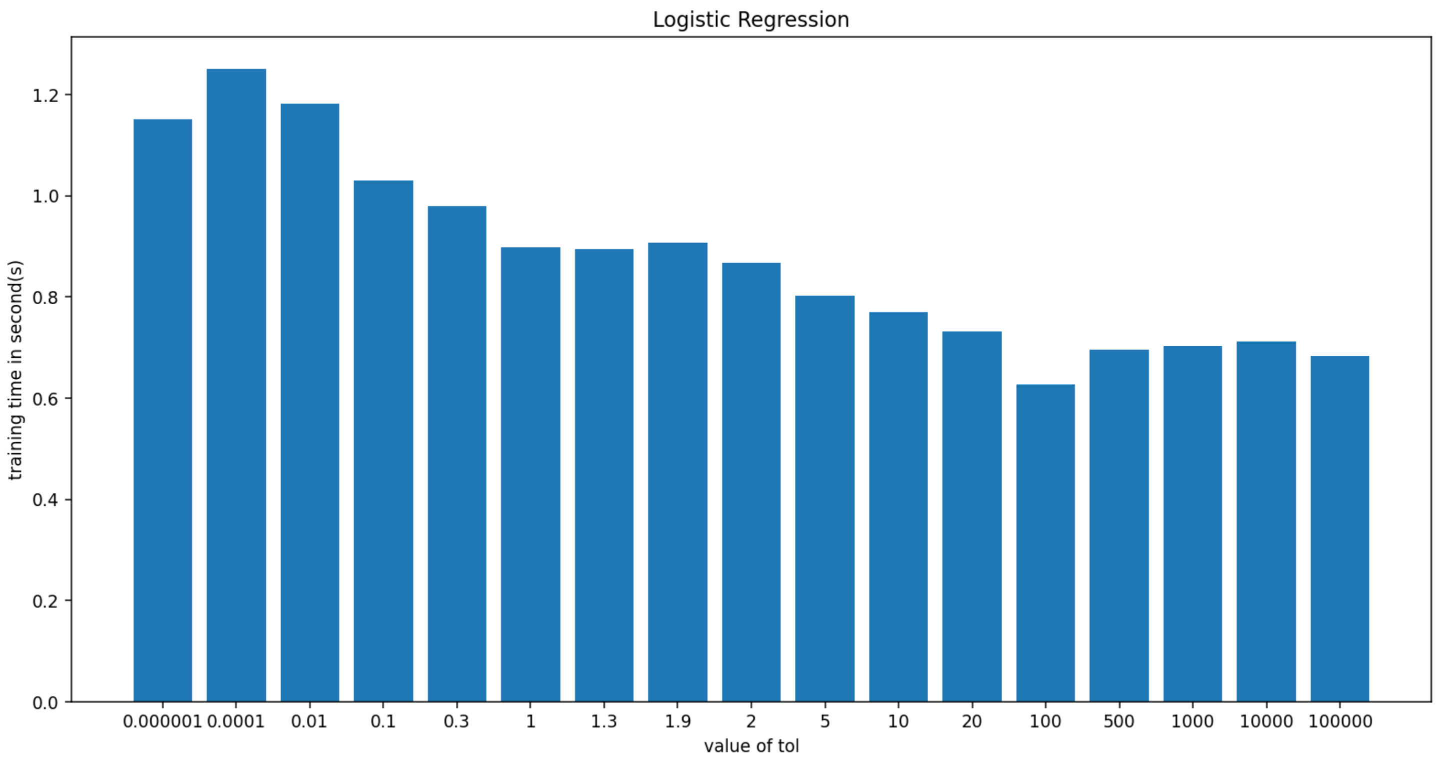


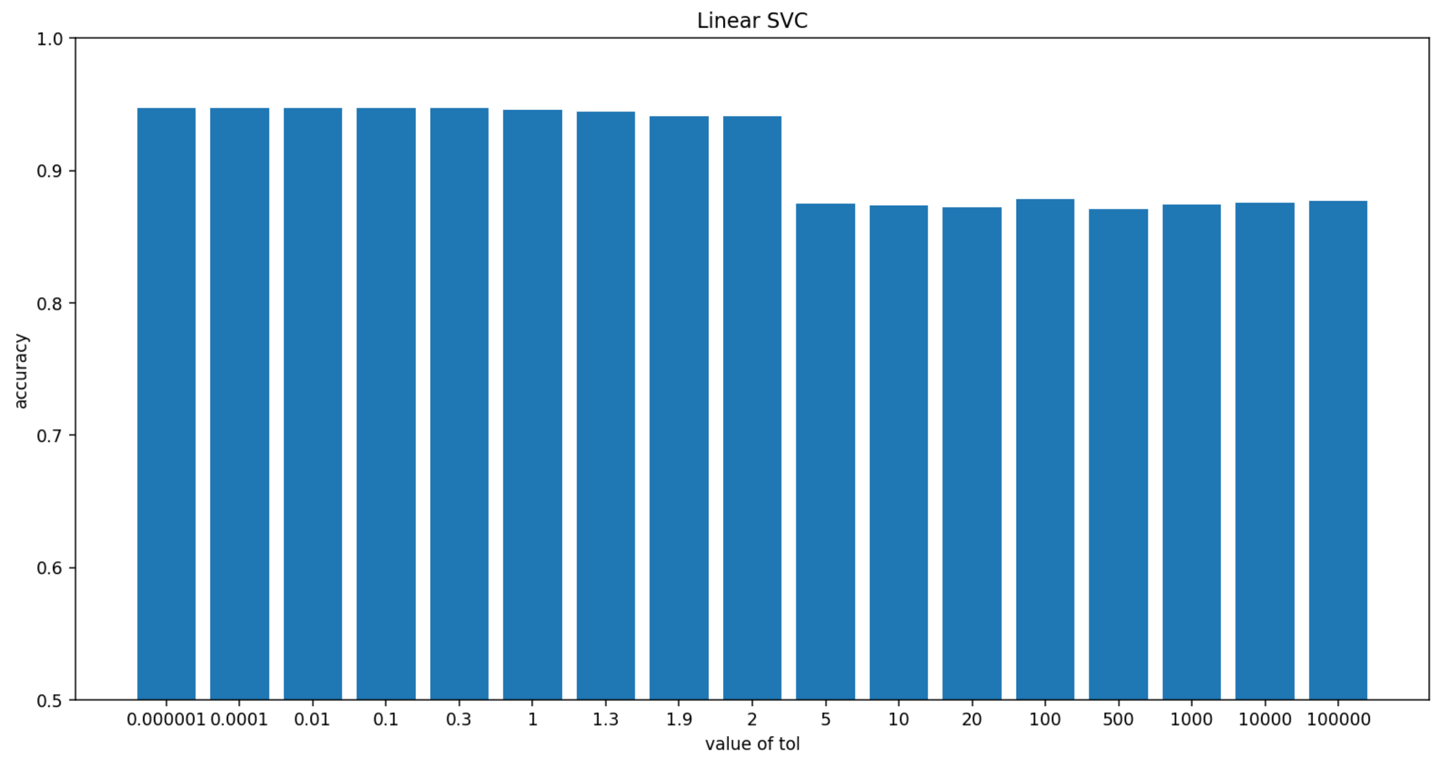

Linear SVC
accuracy
1.0
0.9
0.8
0.7
0.6
0.5
0.000001
0.0001
0.01
0.1
0.3
1
1.3
1.9
2
5
10
20
100
500
1000
10000
100000
value of C

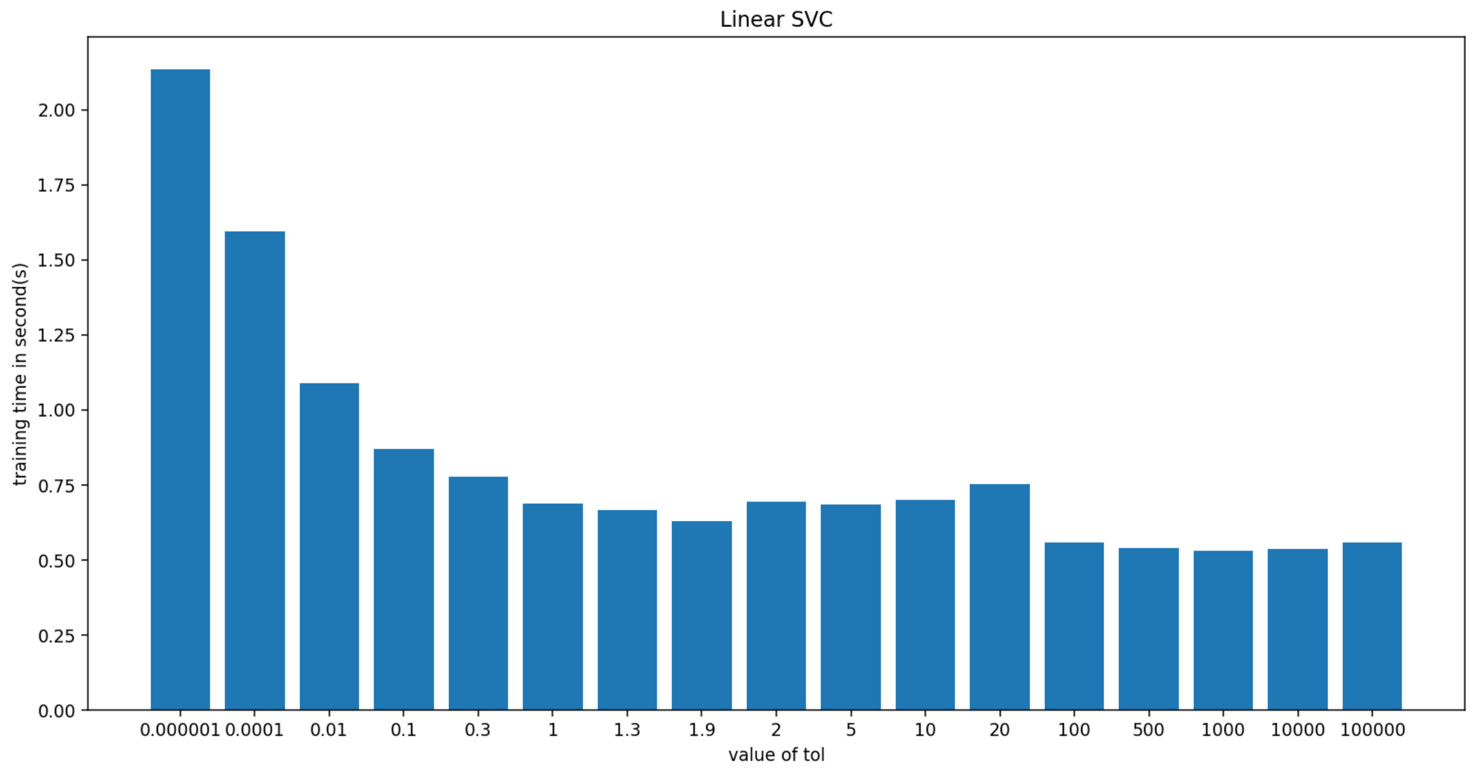

Linear SVC
training time in second(s)
4
3
2
1
0
0.000001
0.0001
0.01
0.1
0.3
1
1.3
1.9
2
5
10
20
100
500
1000
10000
100000
value of C

C) changing the tol hyperparameter in LinearSVC and LogisticRegression






Linear SVC
accuracy
value of tol
1.0
0.9
0.8
0.7
0.6
0.5
0.000001
0.0001
0.01
0.1
0.3
1
1.3
1.9
2
5
10
20
100
500
1000
10000
100000


Linear SVC
training time in second(s)
value of tol
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.000001
0.0001
0.01
0.1
0.3
1
1.3
1.9
2
5
10
20
100
500
1000
10000
100000

D) changing the penalty (regularization) hyperparameter in LinearSVC and LogisticRegression

accuracy for C = 0.3 and tol = 0.316 is 0.9378 and training time is 0.81 seconds for LogisticRegression
As LogisticRegression only support '12'as penalty.So,

for 'l2' : accuracy = 0.9228,training time : 0.7982

accuracy for C = 0.3 and tol = 0.316 is 0.9468 and training time is 0.612 seconds for LinearSVC
LinearSVC support pair of 'l2' and 'squared$_{h}inge'and'dual = True'aspenalty.So$,

for 'l2' : accuracy = 0.9477,training time : 0.5834
so for l2 training time is decreasing and accuracy is increasing by small value.

LinearSVC support pair of 'l2' and 'squared$_{h}inge'and'dual = False'aspenalty.So$,

for 'l2' : accuracy = 0.9367,training time : 0.51
so for l2 training time is decreasing and accuracy is decreasing by small value.

LinearSVC support pair of 'l1' and 'Squared$_{h}inge'and'dual = false'aspenalty.So$,

for 'l1' : accuracy = 0.9315,training time : 0.627
so for l1 training time is decreasing and accuracy is decreasing by small value.