CS771A Assignment 1

The Boys

Rajarshi Dutta 200762 **Udvas Basak** 201056 **Akshit Verma** 200091 **Adit Jain** 200038 **Shivam Pandey** 200938

1 Solution 1

Let us consider the Simple XORRO PUF with two XORROs, each with 64 XOR Gates, as shown in the following Fig (**??**).

As mentioned in the problem statement, the Simple XORRO PUF returns 1, if the upper XORRO has a higher frequency, and returns 0, if the lower XORRO has a higher frequency. Now, the frequency of a XORRO is defined as $f \overset{\text{def}}{=} \frac{1}{t_0+t_1}$, where t_0 is the time taken for the signal to oscillate once when the input is 0, and t_1 is the time taken for the signal to oscillate once when the input is 1. Also, since it is mentioned that the input to any XORRO PUF will have odd number of ones, it is clear that both XORROs will oscillate on every run, and hence, two consecutive runs will have total time $t_0 + t_1$.

Hence, we can imply that, the **Simple XORRO PUF returns 0 if the upper XORRO signal completes two oscillations before the lower XORRO signal, and returns 1 if the lower XORRO signal completes two oscillations before the upper XORRO signal**.

τ_i^u is the (unknown) time after which the signal exits the i^{th} XOR Gate in the upper XORRO PUF, and similarly, τ_i^l is the (unknown) time after which the signal exits the i^{th} XOR Gate in the low XORRO PUF. Note that, the times τ_i^u and τ_i^u depend on the output of the $(i-1)^{th}$ XOR Gate, hence, the times are different for different values of initial inputs into the 0^{th} Gate. Hence, as claimed,

$$\text{Output of Simple XORRO PUF} = \begin{cases} 0, & \text{if } \tau_{63}^u > \tau_{63}^l \\ 1, & \text{if } \tau_{63}^u < \tau_{63}^l \end{cases} \tag{1}$$

Now, we can observe the following:

- If we look at a specific, say i^{th} XOR Gate, for a specific 64-bit input to the XORRO PUF, one input(the one that is the output of the $(i-1)^{th}$ XOR) varies, and the other input remains constant. So essentially, the time taken for the signal to pass that XOR Gate depends on the output of the previous XOR.

- $$\begin{aligned}
A \oplus B &= A \cdot \overline{B} + \overline{A} \cdot B \\
\therefore \overline{A} \oplus B &= \overline{A} \cdot \overline{B} + A \cdot B \\
&= \overline{((\overline{\overline{A} \cdot \overline{B}}) \cdot \overline{(A \cdot B)})} \\
&= \overline{((A + B) \cdot (\overline{A} + \overline{B}))} \\
&= \overline{(A \cdot \overline{A} + B \cdot \overline{A} + A \cdot \overline{B} + B \cdot \overline{B})} \\
&= \overline{(0 + B \cdot \overline{A} + A \cdot \overline{B} + 0)} \\
&= \overline{B \cdot \overline{A} + A \cdot \overline{B}} \\
&= \overline{A \oplus B}
\end{aligned}$$

  which means that, if we invert one input to a XOR Gate, the output also gets inverted. Hence, we can say that, suppose in the first run, where the input to the 0^{th} XOR Gate was 0, if the input to the i^{th} XOR Gate was 0 and to the j^{th} XOR Gate was 1, then in the next run, when the input to the 0^{th} XOR Gate is 1(inverted), all the following XOR Gates will receive inverted inputs, and hence produce inverted outputs, i.e. the input to the i^{th} XOR Gate will be 1 and to the j^{th} XOR Gate will be 0.
- Hence, we can say that, depending on the config bit input, the time taken for the signal to pass through a XOR Gate in two consecutive runs will be:

$$\text{Time Taken for } i^{th} \text{ XOR Gate}(\tau_i) = \begin{cases} \delta^i_{00} + \delta^i_{10}, & \text{if config bit}(a_i) = 0 \\ \delta^i_{01} + \delta^i_{11}, & \text{if config bit}(a_i) = 1 \end{cases} \tag{2}$$

  and hence, we can replace $\delta^i_{00} + \delta^i_{10}$ by $\delta^{i,0}$ and similarly, $\delta^i_{01} + \delta^i_{11}$ by $\delta^{i,1}$.

Therefore,

$$\tau_i = (1 - a_i) \cdot \delta^{i,0} + a_i \cdot \delta^{i,1} + \tau_{i-1} \tag{3}$$

Now, we use the shorthand $\Delta_i \overset{\text{def}}{=} \tau_i^u - \tau_i^l$ to denote the lag between two XOR Gates of the upper and lower XORROs. Note, now that we want to check which signal completes two oscillations first, it has reduced to checking the sign of Δ_{63}.

Hence, the expression for Δ_i becomes,

$$\begin{aligned}
\Delta_i &= (1 - a_i) \cdot (\delta^{i,0,u} - \delta^{i,0,l}) + a_i \cdot (\delta^{i,1,u} - \delta^{i,1,l}) + \Delta_{i-1} \\
\Delta_i &= (1 - a_i) \cdot \delta^0_i + a_i \cdot \delta^1_i + \Delta_{i-1} \qquad [\text{where, } \delta^0_i \overset{\text{def}}{=} \delta^{i,0,u} - \delta^{i,0,l} \text{ and } \delta^1_i \overset{\text{def}}{=} \delta^{i,1,u} - \delta^{i,1,l}] \\
\Delta_i &= a_i(\delta^1_i - \delta^0_i) + \delta^0_i + \Delta_{i-1} \qquad [\Delta_{-1} = 0 \text{ (absorbing all initial delays into } \delta_{ij})]
\end{aligned}$$

And then, expanding,

$$\begin{aligned}
\Delta_0 &= a_0(\delta^1_0 - \delta^0_0) + \delta^0_0 \\
\Delta_1 &= \Delta_0 + a_1(\delta^1_1 - \delta^0_1) + \delta^0_1 \\
&= a_0(\delta^1_0 - \delta^0_0) + \delta^0_0 + a_1(\delta^1_1 - \delta^0_1) + \delta^0_1 \\
&= a_0(\delta^1_0 - \delta^0_0) + a_1(\delta^1_1 - \delta^0_1) + (\delta^0_0 + \delta^0_1) \\
\Delta_2 &= a_0(\delta^1_0 - \delta^0_0) + a_1(\delta^1_1 - \delta^0_1) + a_2(\delta^1_2 - \delta^0_2) + (\delta^0_0 + \delta^0_1 + \delta^0_2) \\
&\vdots \\
\Delta_i &= \sum^i a_i(\delta^1_i - \delta^0_i) + \sum^i \delta^0_i
\end{aligned}$$

Hence the final equation is

$$\Delta_{63} = \omega_0 \cdot x_0 + \omega_1 \cdot x_1 + \ldots + \omega_{63} \cdot x_{63} + \beta_{63} = \boldsymbol{w^T x + b} \tag{4}$$

where,

- $x_i = a_i$
- $w_i = \delta_i^1 - \delta_i^0$ and,
- $b_i = -(\sum \delta_i^0)$
- If $\Delta_{63} < 0$, then output is 0
- If $\Delta_{63} > 0$, then output is 1

Hence, the problem can be correctly responded to by an equation of the form

$$\frac{1 + \text{sign}(w^T \phi(x) + b)}{2}$$

2 Solution 2

From the linear model equated above,

$$\Delta_n = \omega_0 \cdot x_0 + \omega_1 \cdot x_1 + \ldots + \omega_n \cdot x_n + \beta_n = \boldsymbol{w^T x + b} \tag{5}$$

where we find that the model parameters namely the **w** dependent on the difference between the time delays between the **upper** and **lower** XORRO i.e $\delta_i^0 \overset{\text{def}}{=} \delta^{i,0,u} - \delta^{i,0,l}$ and $\delta_i^1 \overset{\text{def}}{=} \delta^{i,1,u} - \delta^{i,1,l}$ and the **bias** term is dependent on the summation of all the individual time delays between the upper and the lower XORROs when each of the config bit a_i is set to 0. Since each of the linear models are distinguishable, a separate linear model is required for a given pair of XORROs (XORRO$_i$ and XORRO$_j$ where i and j are calculated from the first **S** bits and the last **S** bits).

As mentioned in the problem statement, there are 2^s possible XORROs. Each of the two outputs from a pair of XORROs is connected to a $2^{s-1} \times 1$ **Multiplexer**, hence there are $\frac{2^s \cdot (2^s - 1)}{2}$ unordered combinations of pair of XORROs or linear models. According to the problem parameters given, this **Advanced XORRO PUF** can be implemented using **120** linear models. Depending on the upper and lower XORROs selected from the first **S** bits and the last **S** bits, the corresponding linear model is selected from 120 models, and the output is calculated.

It is to be noted here, that while training and testing the model, we have to invert the labels of the entries in the dataset and follow one specific condition. In our case, we are naming the models as $i \rightarrow j$, where $i < j$. So, whenever for this i and j, $i > j$, we have to invert the output label of those rows.

3 Solution 3

The solution of this part can be found in the **submit.py** file in the uploaded zipped file.

4 Solution 4

The given problem has been solved using two Linear models implemented in scikit-learn namely the **Logistic Regression** and **LinearSVC**. The metrics (**accuracy** and **training time**) of the cascaded linear model have been evaluated for different hyperparameters.

4.1 a)

The given table depicts the variation of the accuracy of the **LinearSVC model** with the loss function. The **squared hinge** loss function seems to be a better choice in training the Linear SVM model on the XORRO PUF data.

Loss	Accuracy	Training Time (secs)
Hinge Loss	0.939645	1.9744
Squared Hinge Loss	0.947390	3.5407

4.2 b)

The given tables depict the variation of **training time**, **inference time**, **Accuracy** of the **LinearSVC** and **LogisticRegression** model with change in the C parameter. The plots are also given below for both linear models.

C	Accuracy	Train Time	Test Time
0.01	0.90	1.00	5.33
0.02	0.91	1.78	7.77
0.03	0.92	1.88	7.62
0.04	0.92	1.94	7.95
0.06	0.93	2.06	7.58
0.08	0.93	1.78	7.84
0.1	0.93	2.19	7.92
0.2	0.94	2.55	7.84
0.4	0.94	3.11	7.85
0.6	0.94	3.66	7.77
0.8	0.94	4.00	7.87
1.0	0.94	4.17	7.81
2.0	0.94	3.91	7.74
5.0	0.94	3.64	7.73
10.0	0.94	3.58	7.89
20.0	0.94	3.48	8.03
50.0	0.93	3.60	7.99
75.0	0.93	3.42	7.81
100.0	0.93	3.44	7.73

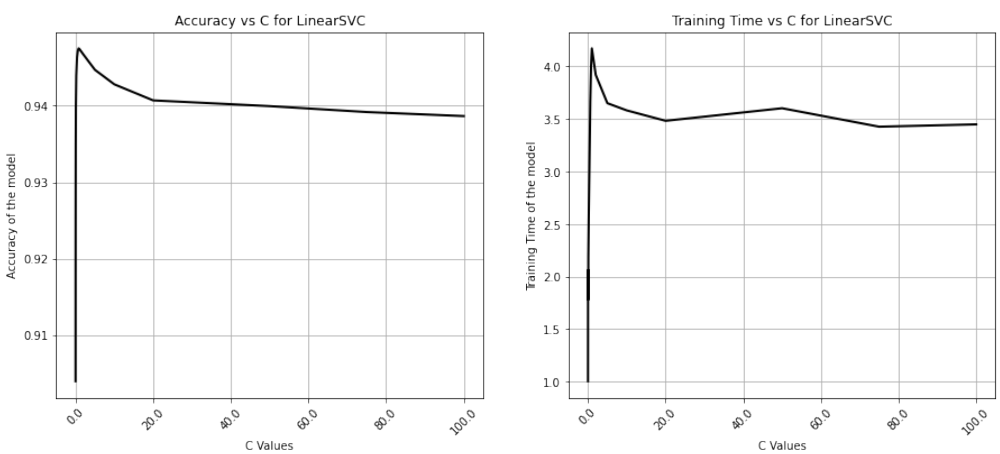

Figure 1: C Characteristics for **LinearSVC** model

C	Accuracy	Train Time	Test Time
0.01	0.8252	2.4519	7.6978
0.02	0.8428	3.9528	7.8315
0.03	0.8554	4.088	7.6928
0.04	0.865	4.3153	7.6293
0.06	0.879	4.3332	7.7207
0.08	0.8897	4.6401	7.795
0.1	0.8963	4.623	7.9073
0.2	0.914	5.1633	7.5939
0.4	0.928	5.5687	7.596
0.6	0.9336	5.8399	7.7969
0.8	0.9372	5.9504	7.8177
1.0	0.9392	6.1105	8.1673
2.0	0.9443	6.8105	7.7491
5.0	0.9485	7.8015	7.7073
10.0	0.9495	8.9107	7.7137
20.0	0.9498	10.2082	7.8481
50.0	0.9497	13.5019	5.1099
75.0	0.9496	5.2057	5.6818
100.0	0.9494	10.5181	6.7603

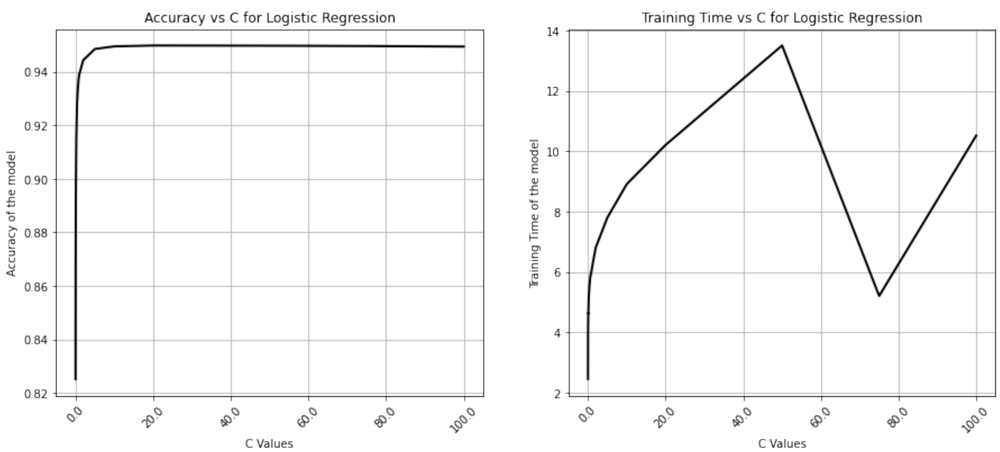

Figure 2: C Characteristics for **Logistic Regression** model

4.3 c)

The given tables depict the variation of **training time**, **inference time**, **Accuracy** of the **LinearSVC** and **LogisticRegression** model with change in the tolerance parameter. The plots are also given below for both linear models.

tol	Accuracy	Train Time	Test Time
1e-5	0.947425	4.0036	7.9229
2e-5	0.947425	4.2789	7.7396
3e-5	0.94735	4.1640	7.7256
4e-5	0.947425	4.0444	7.5732
5e-5	0.947425	3.2668	7.9049
6e-5	0.947425	4.0817	7.7295
7e-5	0.94735	4.1392	7.9170
8e-5	0.947425	4.1288	7.8566
9e-5	0.947425	4.0859	7.8979
10e-5	0.9474	4.1236	7.7015
11e-5	0.9474	4.2288	7.8468
12e-5	0.947325	4.5310	8.4660
13e-5	0.9474	4.1581	7.9033
14e-5	0.947375	4.3758	7.9544
15e-5	0.947425	4.0567	8.0649
16e-5	0.947375	3.9643	6.1963
17e-5	0.947425	4.2655	7.8078
18e-5	0.94735	3.8480	7.9810
19e-5	0.94735	4.1530	8.0560
20e-5	0.947375	3.9634	7.7918

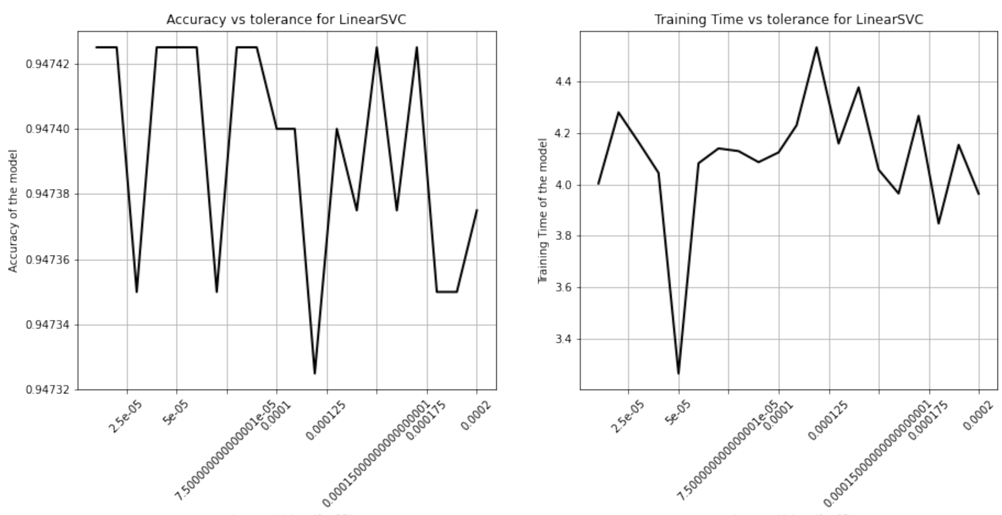

Figure 3: Tolerance Characteristics for **LinearSVC** model

tol	Accuracy	Train Time	Test Time
1e-5	0.939	1.79	2.79
2e-5	0.939	1.58	2.76
3e-5	0.939	1.64	2.80
4e-5	0.939	1.58	2.78
5e-5	0.939	1.65	2.75
6e-5	0.939	1.58	2.78
7e-5	0.939	1.58	2.77
8e-5	0.939	1.63	2.81
9e-5	0.939	1.68	3.014
10e-5	0.939	1.71	2.76
11e-5	0.939	1.59	2.90
12e-5	0.939	1.69	3.02
13e-5	0.939	1.73	2.92
14e-5	0.939	1.65	2.78
15e-5	0.939	1.58	2.85
16e-5	0.939	1.60	5.72
17e-5	0.939	5.32	2.89
18e-5	0.939	1.60	2.85
19e-5	0.939	1.61	2.86
20e-5	0.939	1.62	2.83

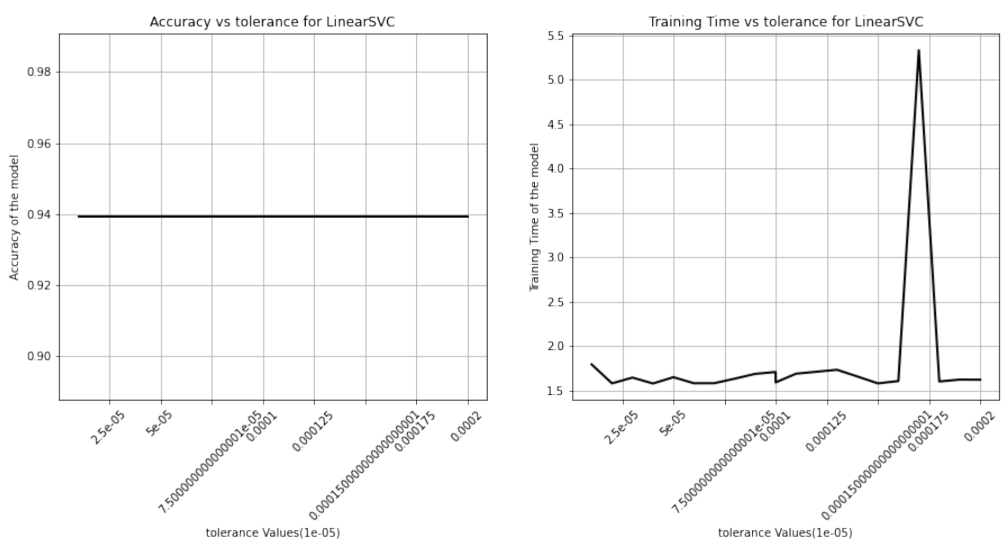

Figure 4: Tolerance Characteristics for **Logistic Regression** model

The next set of plots depicts the variation of the accuracy of the **LinearSVC** and the **LogisticRegression** model with different ranges of the **max_iter** parameter.

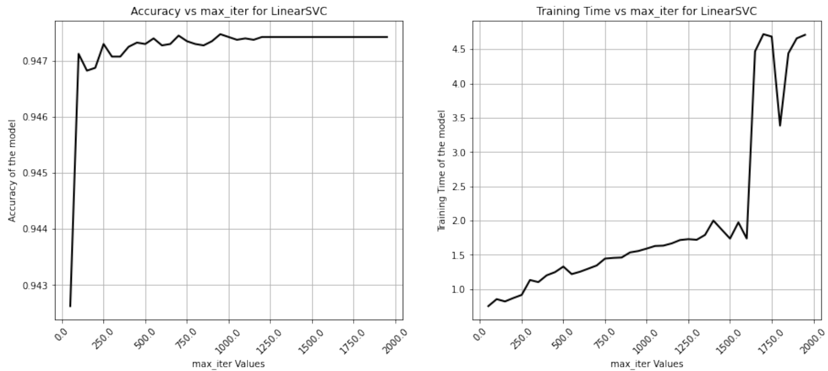

Figure 5: Max iter Characteristics for **LinearSVC** model in the range 50-200

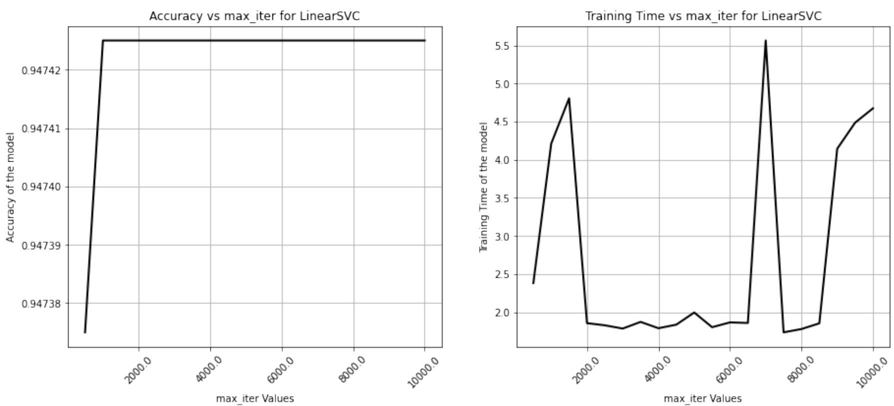

Figure 6: Max iter Characteristics for **LinearSVC** model in the range 500-10000

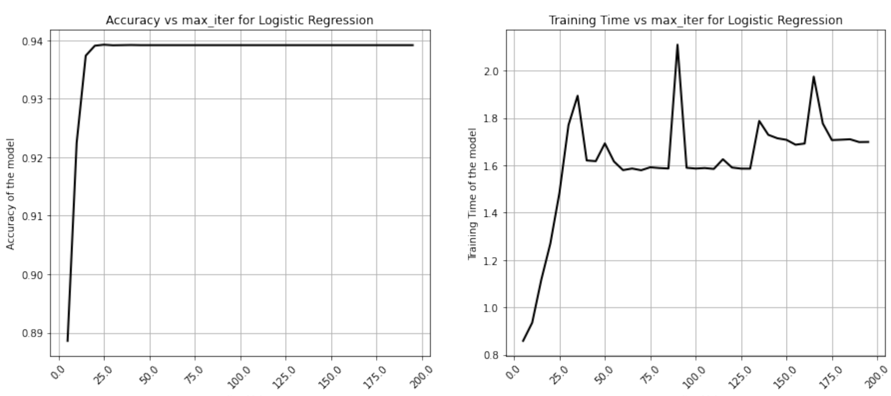

Figure 7: Max iter Characteristics for **Logistic Regression** model in 5-200 range

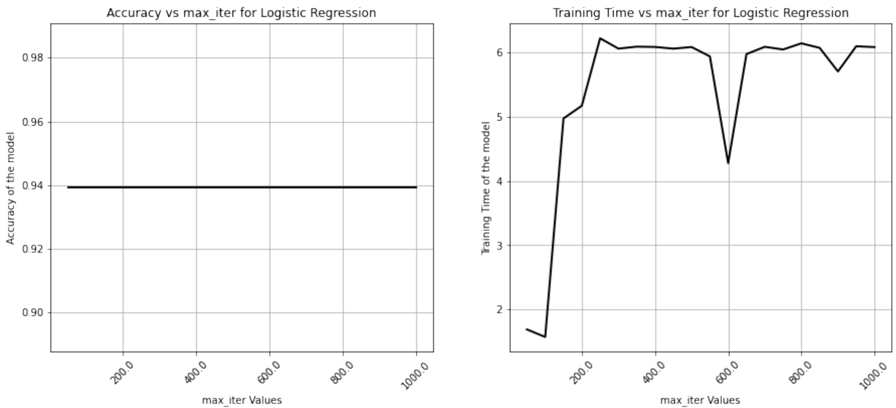

Figure 8: Max iter Characteristics for **Logistic Regression** model in 200-1000 range

4.4 d)

The given tables depict the variation of the accuracy of the **Logistic Regression** and **LinearSVC** model with change in the penalty/regularization (**l1** and **l2** type).

Penalty	Accuracy	Training Time(secs)
L1	0.94596	13.07
L2	0.94741	3.55

(a) LinearSVC model

Penalty	Accuracy	Training Time(secs)
L1	0.93534	1.03
L2	0.93917	0.81

(b) Logistic Regression model

4.5 Discussion

It is evident that the best model would be a **Logistic Regression Model** with L2 penalty, C value of 20.0, tolerance value = 8e-5 and max_iter = 150. The accuracy of this model on the XORRO PUF dataset comes out to be 94.98 %.