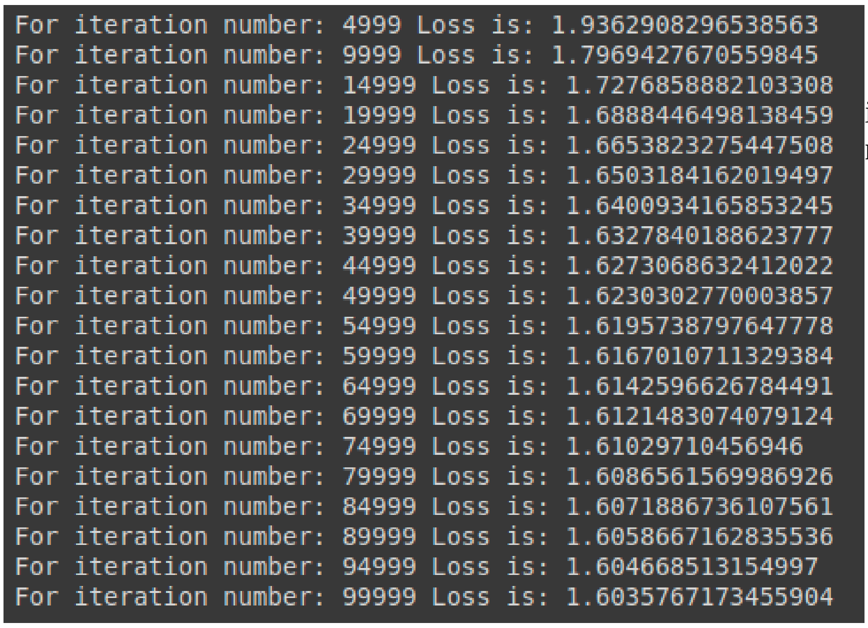
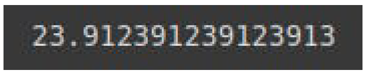
Linear Regression:

Linear reg was the easiest one to implement among the 4 models,however still gave poorest result of all. Initially raised a doubt in me,if the program written itself was inefficient or not;but later realised applying a regression model on a classification problem isn't a great idea. Not to mention,tuning the parameters was not of much help and this stuff ended with an accuracy of mere **24 %**.



```
For iteration number: 4999 Loss is: 1.9362908296538563
For iteration number: 9999 Loss is: 1.7969427670559845
For iteration number: 14999 Loss is: 1.7276858882103308
For iteration number: 19999 Loss is: 1.6888446498138459
For iteration number: 24999 Loss is: 1.6653823275447508
For iteration number: 29999 Loss is: 1.6503184162019497
For iteration number: 34999 Loss is: 1.6400934165853245
For iteration number: 39999 Loss is: 1.6327840188623777
For iteration number: 44999 Loss is: 1.6273068632412022
For iteration number: 49999 Loss is: 1.6230302770003857
For iteration number: 54999 Loss is: 1.6195738797647778
For iteration number: 59999 Loss is: 1.6167010711329384
For iteration number: 64999 Loss is: 1.6142596626784491
For iteration number: 69999 Loss is: 1.6121483074079124
For iteration number: 74999 Loss is: 1.61029710456946
For iteration number: 79999 Loss is: 1.6086561569986926
For iteration number: 84999 Loss is: 1.6071886736107561
For iteration number: 89999 Loss is: 1.6058667162835536
For iteration number: 94999 Loss is: 1.604668513154997
For iteration number: 99999 Loss is: 1.6035767173455904
```

Cost at every 5000 iters(screenshot from colab notebook)



```
23.912391239123913
```

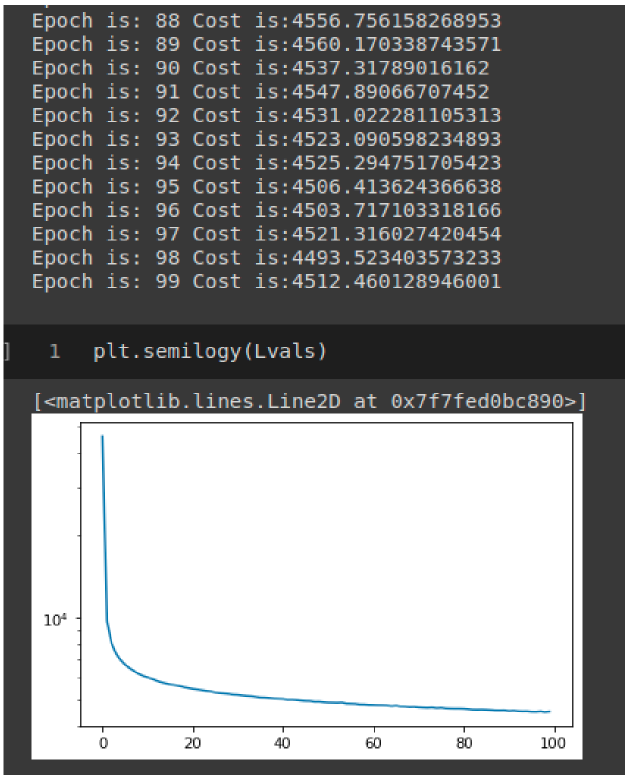
Final accuracy(screenshot from colab notebook)

Logistic Regression:

This was the hardest one to code and took most of the time to get executed,however it returned pretty great results and so was worth putting effort into. The idea of finding probabilities of a dataset being of a particular label, in itself, is so meaningful and surely got my interest on.

However, messed up this fellow a lot and every second line generated a shape error, However several blogs and youtube videos did come to help.(Here is where I realised how important it is to put comments to make the action of each line understandable.)

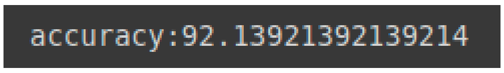
The most noticable thing was, the accuracy was more sensible to learning rate than any other parameters. Tuning it was the most tideous job.An epoch of 10 gave around 90.8%, while 100 epochs returned **92.15%**.In all,things broke a lot in the middle,but it was fun altogether: pretty decent results added to it.



```
Epoch is: 88 Cost is:4556.756158268953
Epoch is: 89 Cost is:4560.170338743571
Epoch is: 90 Cost is:4537.31789016162
Epoch is: 91 Cost is:4547.89066707452
Epoch is: 92 Cost is:4531.022281105313
Epoch is: 93 Cost is:4523.090598234893
Epoch is: 94 Cost is:4525.294751705423
Epoch is: 95 Cost is:4506.413624366638
Epoch is: 96 Cost is:4503.717103318166
Epoch is: 97 Cost is:4521.316027420454
Epoch is: 98 Cost is:4493.523403573233
Epoch is: 99 Cost is:4512.460128946001
```

```
plt.semilogy(Lvals)
```

```
[<matplotlib.lines.Line2D at 0x7f7fed0bc890>]
```

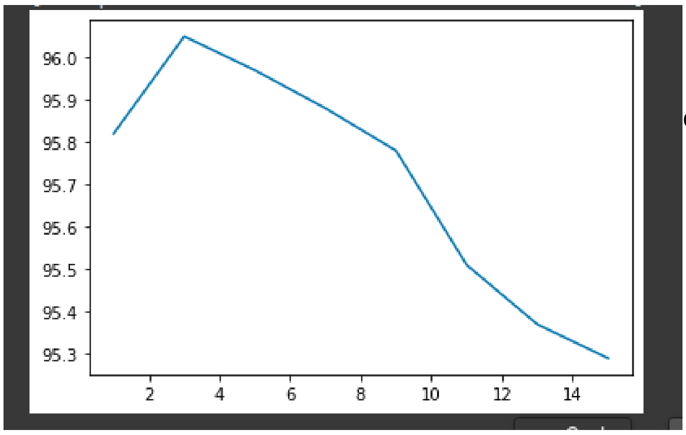


```
accuracy:92.13921392139214
```

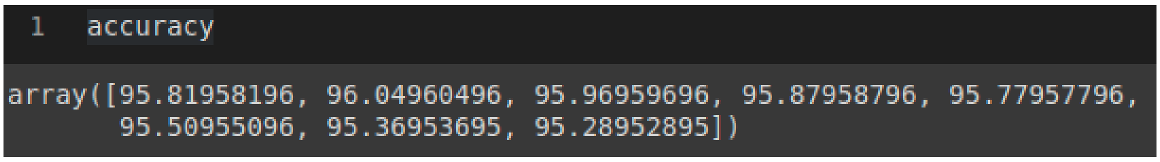
Cost of last few epochs and accuracy(screenshot from colab notebook)

KNN:

It took the least time to implement,the algorithm too was quite easy. Experimented by varying code a lot. Not gonna lie, took ideas from a lot of blogs and articles , implemented some recently learned cool tricks. K=3 gave the best accuracy of **96%**.



Accuracy vs values of K(screenshot from colab notebook)



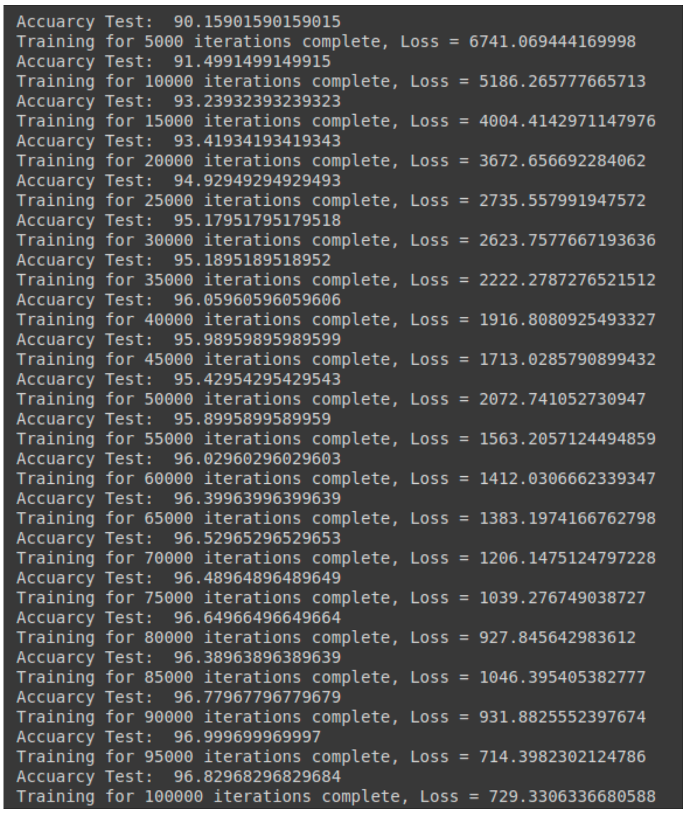
```
accuracy
```

```
array([95.81958196, 96.04960496, 95.96959696, 95.87958796, 95.77957796,
       95.50955096, 95.36953695, 95.28952895])
```

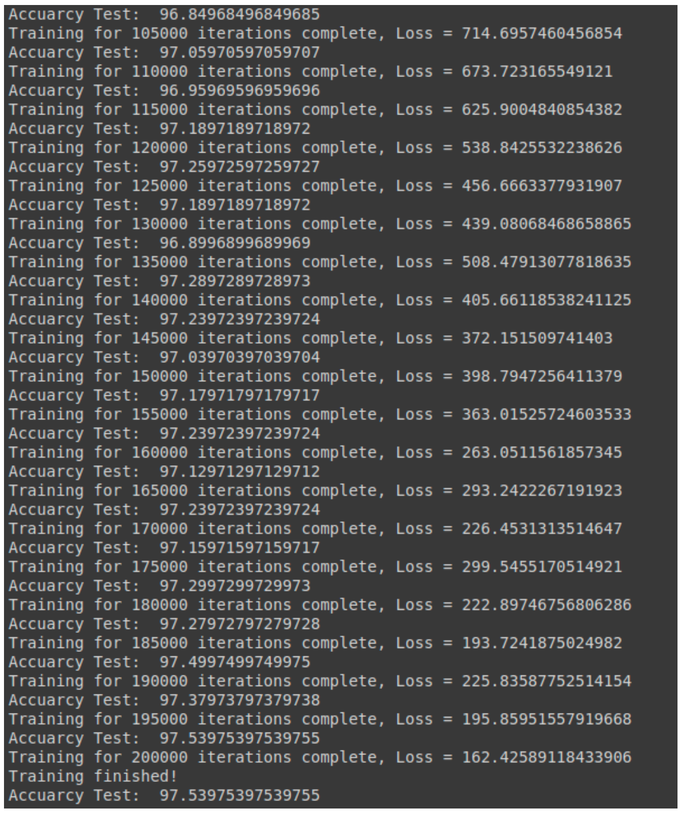
Accuracy Array(screenshot from colab notebook)

Neural Network:

Neural networking was the most exciting part of the project. Took a lot of time to go through the underlying mathematics, but worth the time.Gave the best result out of all the 4 models.
A considerable difference in the outcomes was noticeable based on the combination of activation functions used(ReLU on layer 1 and softmax on layer 2 worked the best) and how the initial weights and biases were initialised.
Iterations for 2 lakh times caused an accuracy of **97.54%**; however, the continuous decreasing trend in the losses suggests even more iterations can be done to attain higher accuracy.



```
Accuarcy Test:  90.15901590159015
Training for 5000 iterations complete, Loss = 6741.069444169998
Accuarcy Test:  91.4991499149915
Training for 10000 iterations complete, Loss = 5186.265777665713
Accuarcy Test:  93.23932393239323
Training for 15000 iterations complete, Loss = 4004.4142971147976
Accuarcy Test:  93.41934193419343
Training for 20000 iterations complete, Loss = 3672.656692284062
Accuarcy Test:  94.92949294929493
Training for 25000 iterations complete, Loss = 2735.557991947572
Accuarcy Test:  95.17951795179518
Training for 30000 iterations complete, Loss = 2623.7577667193636
Accuarcy Test:  95.1895189518952
Training for 35000 iterations complete, Loss = 2222.2787276521512
Accuarcy Test:  96.05960596059606
Training for 40000 iterations complete, Loss = 1916.8080925493327
Accuarcy Test:  95.98959895989599
Training for 45000 iterations complete, Loss = 1713.0285790899432
Accuarcy Test:  95.42954295429543
Training for 50000 iterations complete, Loss = 2072.741052730947
Accuarcy Test:  95.8995899589959
Training for 55000 iterations complete, Loss = 1563.2057124494859
Accuarcy Test:  96.02960296029603
Training for 60000 iterations complete, Loss = 1412.0306662339347
Accuarcy Test:  96.39963996399639
Training for 65000 iterations complete, Loss = 1383.1974166762798
Accuarcy Test:  96.52965296529653
Training for 70000 iterations complete, Loss = 1206.1475124797228
Accuarcy Test:  96.48964896489649
Training for 75000 iterations complete, Loss = 1039.276749038727
Accuarcy Test:  96.64966496649664
Training for 80000 iterations complete, Loss = 927.845642983612
Accuarcy Test:  96.38963896389639
Training for 85000 iterations complete, Loss = 1046.395405382777
Accuarcy Test:  96.77967796779679
Training for 90000 iterations complete, Loss = 931.8825552397674
Accuarcy Test:  96.999699969997
Training for 95000 iterations complete, Loss = 714.3982302124786
Accuarcy Test:  96.82968296829684
Training for 100000 iterations complete, Loss = 729.3306336680588
```



```
Accuarcy Test:  96.84968496849685
Training for 105000 iterations complete, Loss = 714.6957460456854
Accuarcy Test:  97.05970597059707
Training for 110000 iterations complete, Loss = 673.723165549121
Accuarcy Test:  96.95969596959696
Training for 115000 iterations complete, Loss = 625.9004840854382
Accuarcy Test:  97.1897189718972
Training for 120000 iterations complete, Loss = 538.8425532238626
Accuarcy Test:  97.25972597259727
Training for 125000 iterations complete, Loss = 456.6663377931907
Accuarcy Test:  97.1897189718972
Training for 130000 iterations complete, Loss = 439.08068468658865
Accuarcy Test:  96.8996899689969
Training for 135000 iterations complete, Loss = 508.47913077818635
Accuarcy Test:  97.2897289728973
Training for 140000 iterations complete, Loss = 405.66118538241125
Accuarcy Test:  97.23972397239724
Training for 145000 iterations complete, Loss = 372.151509741403
Accuarcy Test:  97.03970397039704
Training for 150000 iterations complete, Loss = 398.7947256411379
Accuarcy Test:  97.17971797179717
Training for 155000 iterations complete, Loss = 363.01525724603533
Accuarcy Test:  97.23972397239724
Training for 160000 iterations complete, Loss = 263.0511561857345
Accuarcy Test:  97.12971297129712
Training for 165000 iterations complete, Loss = 293.2422267191923
Accuarcy Test:  97.23972397239724
Training for 170000 iterations complete, Loss = 226.4531313514647
Accuarcy Test:  97.15971597159717
Training for 175000 iterations complete, Loss = 299.5455170514921
Accuarcy Test:  97.2997299729973
Training for 180000 iterations complete, Loss = 222.89746756806286
Accuarcy Test:  97.27972797279728
Training for 185000 iterations complete, Loss = 193.7241875024982
Accuarcy Test:  97.4997499749975
Training for 190000 iterations complete, Loss = 225.83587752514154
Accuarcy Test:  97.37973797379738
Training for 195000 iterations complete, Loss = 195.85951557919668
Accuarcy Test:  97.53975397539755
Training for 200000 iterations complete, Loss = 162.42589118433906
Training finished!
Accuarcy Test:  97.53975397539755
```

Final accuracy and loss at every 5000 iterations(screenshot from Colab notebook)