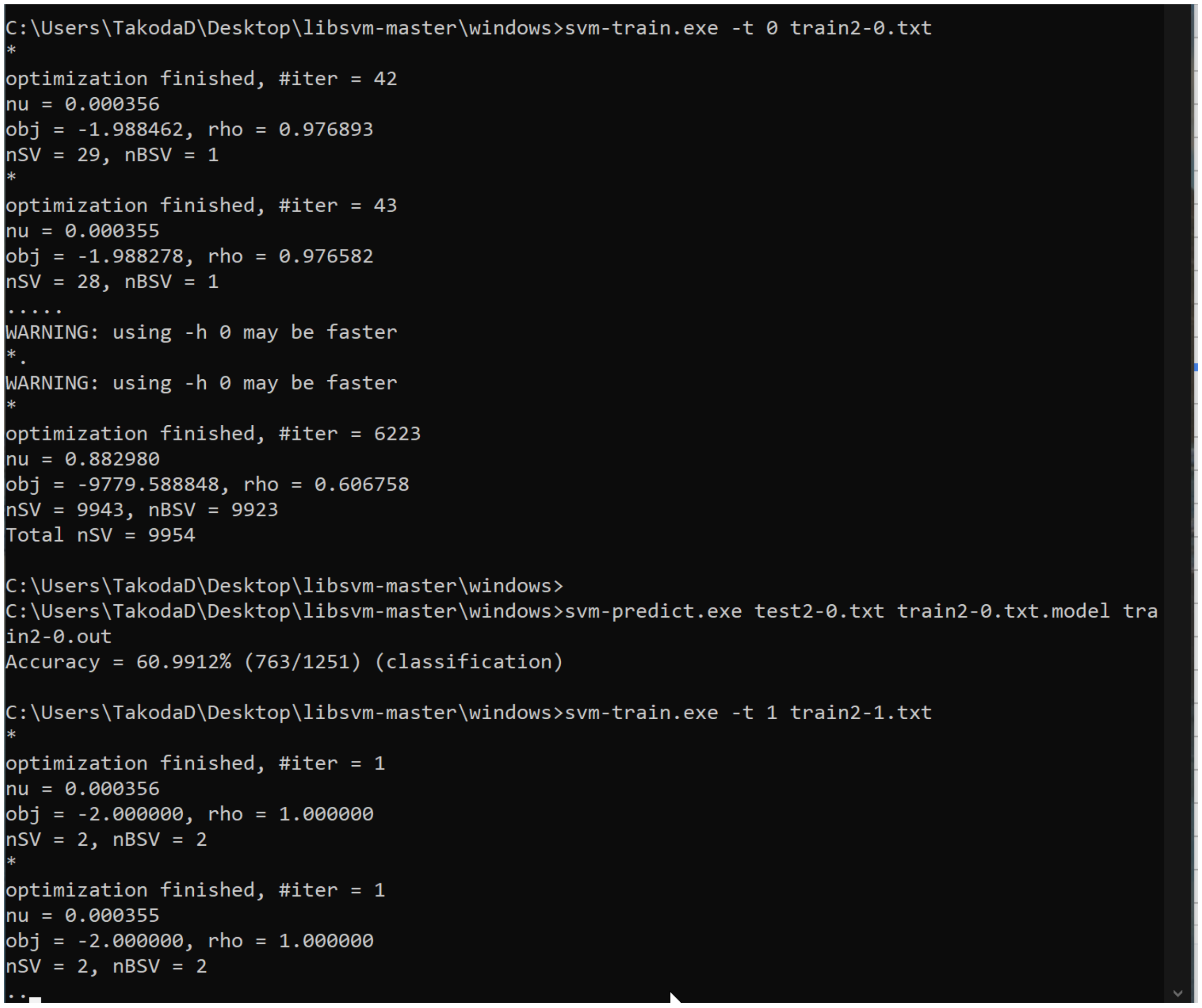
Example of how data was obtained:

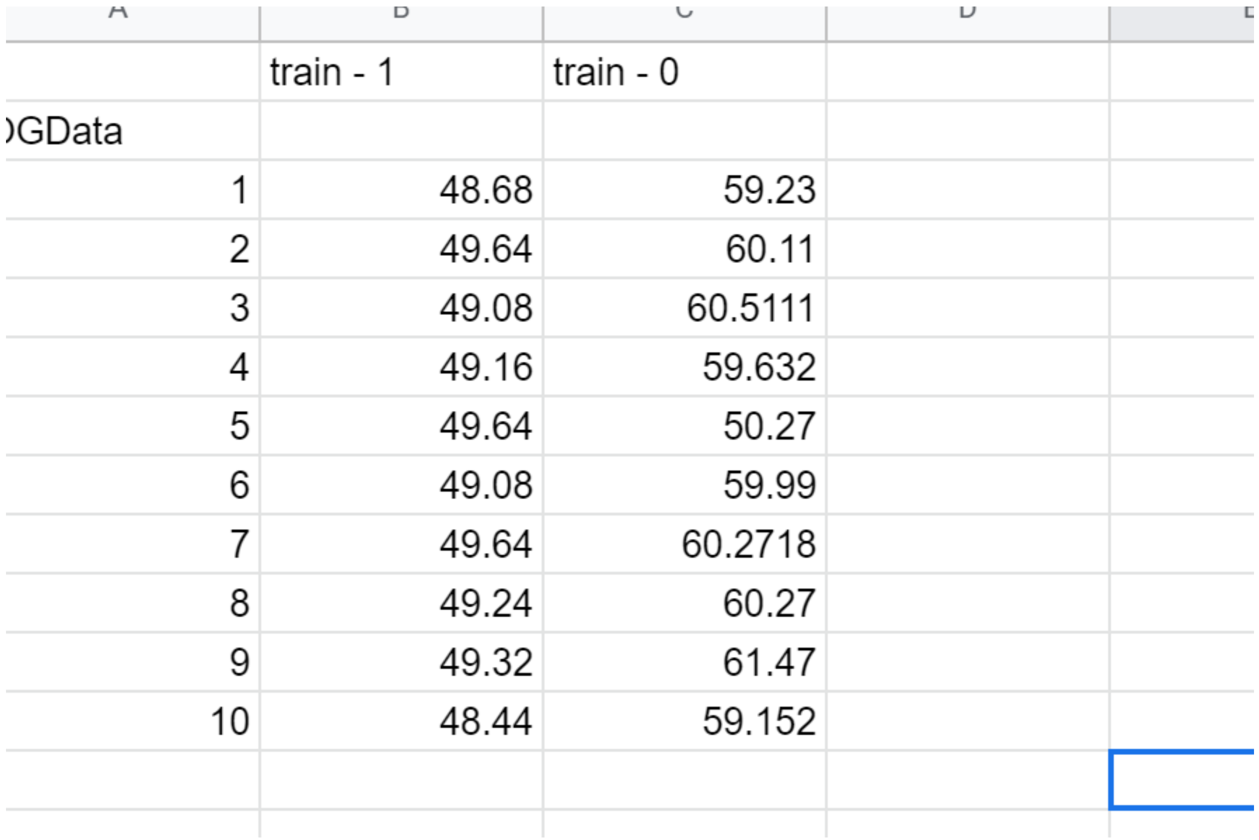


```
C:\Users\TakodaD\Desktop\libsvm-master\windows>svm-train.exe -t 0 train2-0.txt
*
optimization finished, #iter = 42
nu = 0.000356
obj = -1.988462, rho = 0.976893
nSV = 29, nBSV = 1
*
optimization finished, #iter = 43
nu = 0.000355
obj = -1.988278, rho = 0.976582
nSV = 28, nBSV = 1
.....
WARNING: using -h 0 may be faster
*.
WARNING: using -h 0 may be faster
*
optimization finished, #iter = 6223
nu = 0.882980
obj = -9779.588848, rho = 0.606758
nSV = 9943, nBSV = 9923
Total nSV = 9954

C:\Users\TakodaD\Desktop\libsvm-master\windows>
C:\Users\TakodaD\Desktop\libsvm-master\windows>svm-predict.exe test2-0.txt train2-0.txt.model tra
in2-0.out
Accuracy = 60.9912% (763/1251) (classification)

C:\Users\TakodaD\Desktop\libsvm-master\windows>svm-train.exe -t 1 train2-1.txt
*
optimization finished, #iter = 1
nu = 0.000356
obj = -2.000000, rho = 1.000000
nSV = 2, nBSV = 2
*
optimization finished, #iter = 1
nu = 0.000355
obj = -2.000000, rho = 1.000000
nSV = 2, nBSV = 2
..
```

Data:



| A | B | C | D |
|---|---|---|---|
| | train - 1 | train - 0 | |
| GData | | | |
| 1 | 48.68 | 59.23 | |
| 2 | 49.64 | 60.11 | |
| 3 | 49.08 | 60.5111 | |
| 4 | 49.16 | 59.632 | |
| 5 | 49.64 | 50.27 | |
| 6 | 49.08 | 59.99 | |
| 7 | 49.64 | 60.2718 | |
| 8 | 49.24 | 60.27 | |
| 9 | 49.32 | 61.47 | |
| 10 | 48.44 | 59.152 | |

**Conclusion:**
Based on the trials (1-3), the solution I will be going with is a linear kernel with adaboost algorithm. The Linear Kernel (59% avg) outperformed the Polynomial Kernel(49% avg) by an average of 10%. I initially expected the polynomial kernel to outperform the linear kernel, however, was shocked to see other results unfold. With the addition of the adaboost algorithm, the linear kernel is the clear solution to this problem.