CSCI 5525 Machine Learning
Assignment 2

Ritiz Tambi
tambi004

Using 1 Grace Day

Problem 2 partner - Krishna Chaitanya Bandi (bandi014)

Problem 1

A)

Lagrange Dual for a non-separable soft-margin case:

$$L^*(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j x_i x_j$$

To fit the above equation for CVXOPT

$\max(1^T\alpha - \frac{1}{2}\alpha^T K\alpha)$ => $\min(\frac{1}{2}\alpha^T K\alpha - 1^T\alpha)$

Constraints:

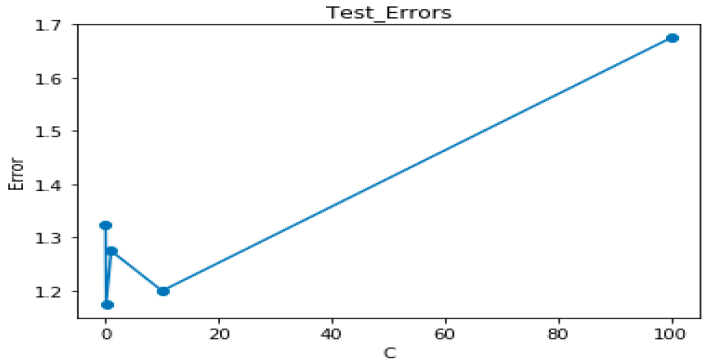
$\alpha \geq 0$, $-\alpha \geq -C$, $\sum_i \alpha_i y_i = 0$

B)

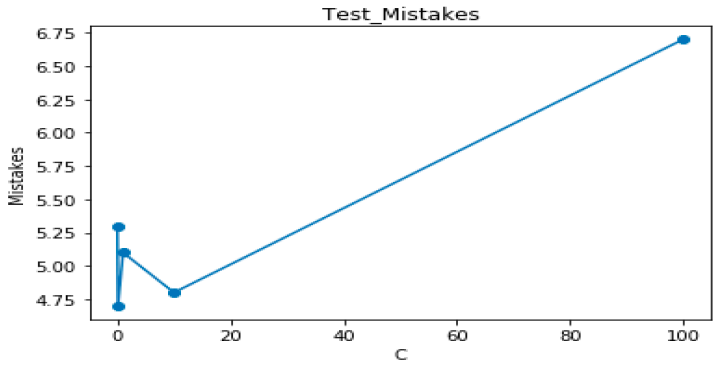
(i)

C	Run_1	Run_2	Run_3	Run_4	Run_5	Run_6	Run_7	Run_8	Run_9	Run_10
0.01	1.5	0.5	0.75	1.25	1.75	1.75	1	2.25	1.25	1.25
0.1	1.75	0.75	0.5	2.0	0.75	0.75	1.25	1.75	1.5	0.75
1	1.25	2	1.75	0.5	0.75	1.25	1.5	2	1.25	0.5
10	1.25	0.5	0.75	1.5	1.25	2	0.75	0.75	2.25	1.25
100	2.25	0.75	1.25	2	1	1.5	1.5	3	1.5	1.75

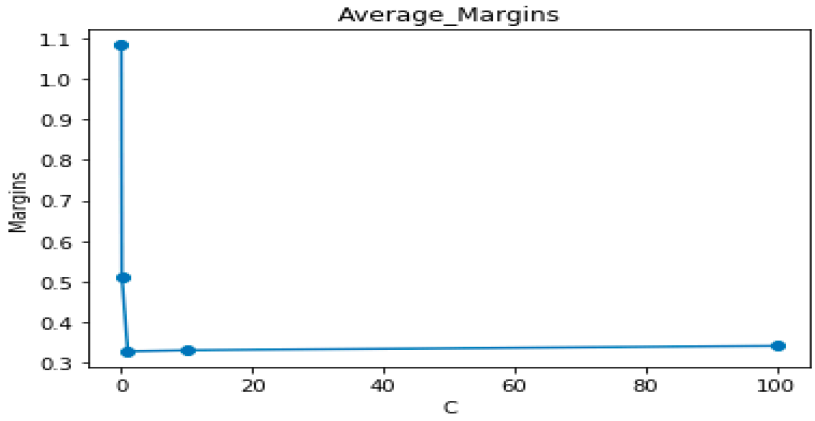
```
Summary and Plots
----------------
Average Test_Errors:
C: 0.01 Test_error: 1.3249999999999975 Standard_Deviation: 0.4879805323985777
C: 0.1 Test_error: 1.174999999999997 Standard_Deviation: 0.5129571132170807
C: 1 Test_error: 1.2749999999999981 Standard_Deviation: 0.5297405025104278
C: 10 Test_error: 1.1999999999999975 Standard_Deviation: 0.5678908345800276
C: 100 Test_error: 1.6749999999999996 Standard_Deviation: 0.6025985396596982
```



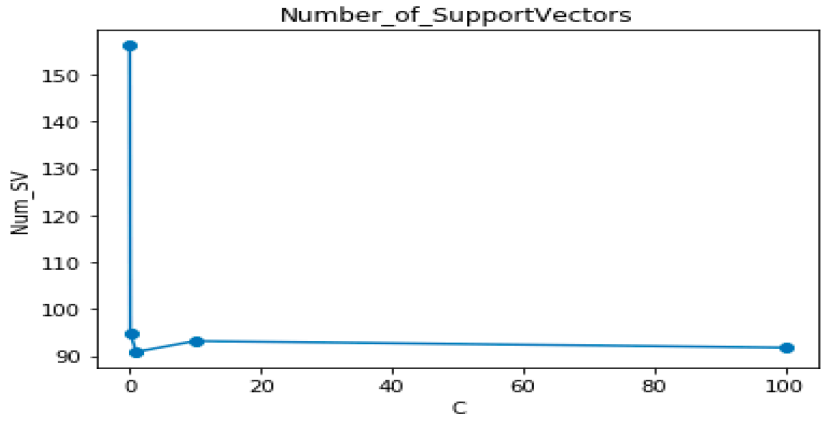

```
----------------
Average Test_Mistakes:
C: 0.01 Test_Mistakes: 5.3
C: 0.1 Test_Mistakes: 4.7
C: 1 Test_Mistakes: 5.1
C: 10 Test_Mistakes: 4.8
C: 100 Test_Mistakes: 6.7
```



```
_________________
Average Margins:
C: 0.01 Average_Margin: 1.082941963538976
C: 0.1 Average_Margin: 0.5128931427539817
C: 1 Average_Margin: 0.3288795995815271
C: 10 Average_Margin: 0.33156337288838184
C: 100 Average_Margin: 0.34237794856886566
```




```
_________________
Average Number of Support Vectors:
C: 0.01 Average_Num_SupportVectors: 156.2
C: 0.1 Average_Num_SupportVectors: 94.8
C: 1 Average_Num_SupportVectors: 90.9
C: 10 Average_Num_SupportVectors: 93.2
C: 100 Average_Num_SupportVectors: 91.8
```



(ii)

```
Average Test_Errors:
C: 0.01 Test_error: 1.3249999999999975 Standard_Deviation: 0.4879805323985777
C: 0.1 Test_error: 1.174999999999997 Standard_Deviation: 0.5129571132170807
C: 1 Test_error: 1.2749999999999981 Standard_Deviation: 0.5297405025104278
C: 10 Test_error: 1.1999999999999975 Standard_Deviation: 0.5678908345800276
C: 100 Test_error: 1.6749999999999996 Standard_Deviation: 0.6025985396596982
```

Initially as C increases test-error decreases. With further increase in C values, test-error also increases as depicted by the test-error values and plots above.

(iii)

```
Average Margins:
C: 0.01 Average_Margin: 1.082941963538976
C: 0.1 Average_Margin: 0.5128931427539817
C: 1 Average_Margin: 0.3288795995815271
C: 10 Average_Margin: 0.33156337288838184
C: 100 Average_Margin: 0.34237794856886566
```

As depicted above by the plots and average geometric-margin values per C; with increase in C values geometric margins decrease

(iv)

```
Average Number of Support Vectors:
C: 0.01 Average_Num_SupportVectors: 156.2
C: 0.1 Average_Num_SupportVectors: 94.8
C: 1 Average_Num_SupportVectors: 90.9
C: 10 Average_Num_SupportVectors: 93.2
C: 100 Average_Num_SupportVectors: 91.8
```

As depicted above by the plots and average number of Support Vector values per C; with increase in C values number of support vectors decrease. This is due to decreasing number of points within the margins as margins value decrease with C.

C)

c)

i) α values increase with increase in C.

$w = \sum_i \alpha_i x_i y_i \Rightarrow$ margin $= 1/\|w\|$ decreases with C ↑

Hence this increasing margin can lead to misclassification affecting accuracy of the model on test examples.

Hence choosing the right C value is important

ii) Given

$$\frac{1}{2}\|w\|^2 + C \sum_i \xi_i(w, b)$$

$$\xi_i = \xi_i(w, b)$$

$$y_i f(x_i) \geq 1 - \xi_i$$

$$\Rightarrow \quad \xi_i \geq 1 - y_i f(x_i) \qquad -(1)$$

$$\xi_i \geq \qquad - \qquad (2)$$

$$\Rightarrow \quad \xi_i = \max(1 - y_i f(x_i), 0)$$

$$\xi_i = \max(1 - y_i(w^T x_i + b), 0)$$

⇓

Hinge Loss

This eqn satisfies all the constraints.

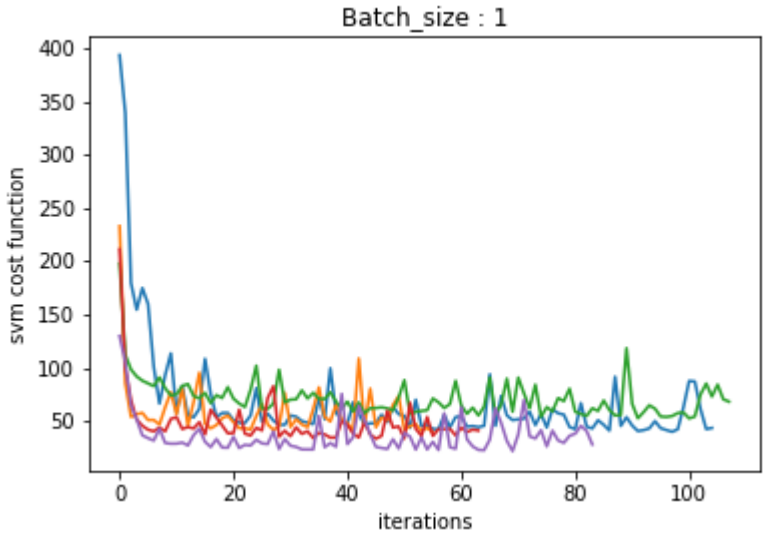
Problem 2

a)

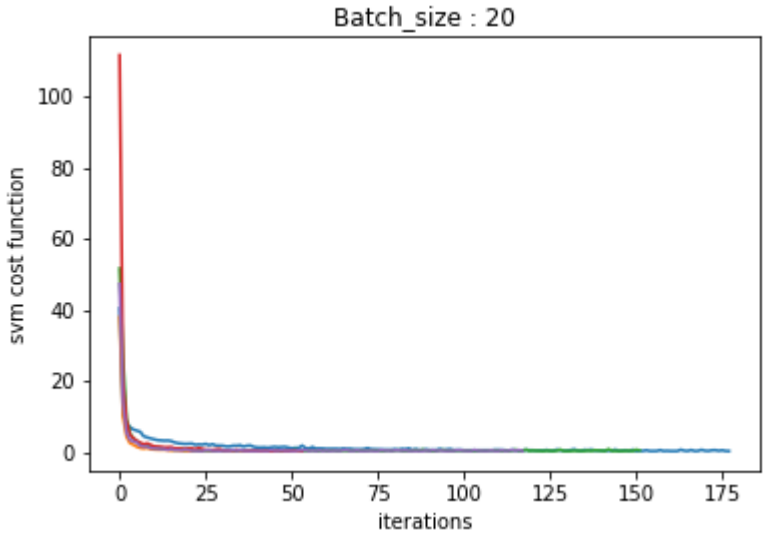
Implementing the **Pegasos algorithm** stated in the Research Paper, we obtained the following mean and standard deviation of runtime results. We observe that the loss function value decreases rapidly with increasing number of iterations.

Note: Our graphs depict for epochs till 400, but loss decreases further till n_epochs = 2000

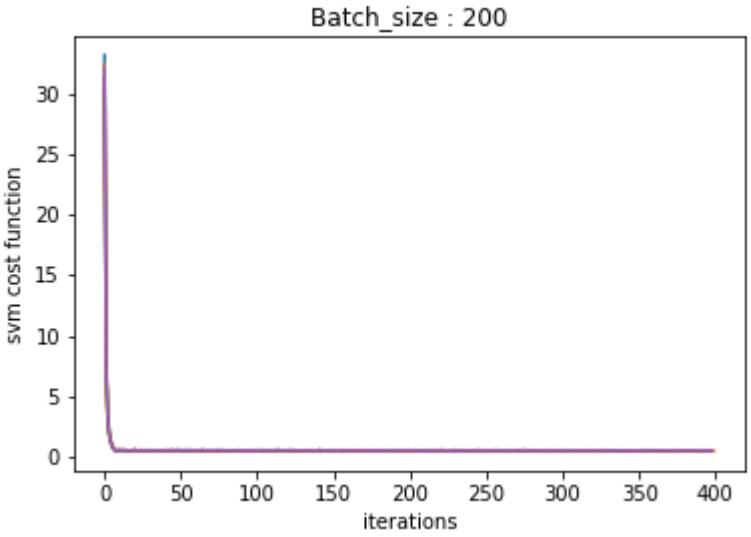
Batch Size (k) = 1
Number of runs = 5
Mean Runtime = 0.0540613174 seconds
Standard Deviation Runtime = 0.0054093021 seconds



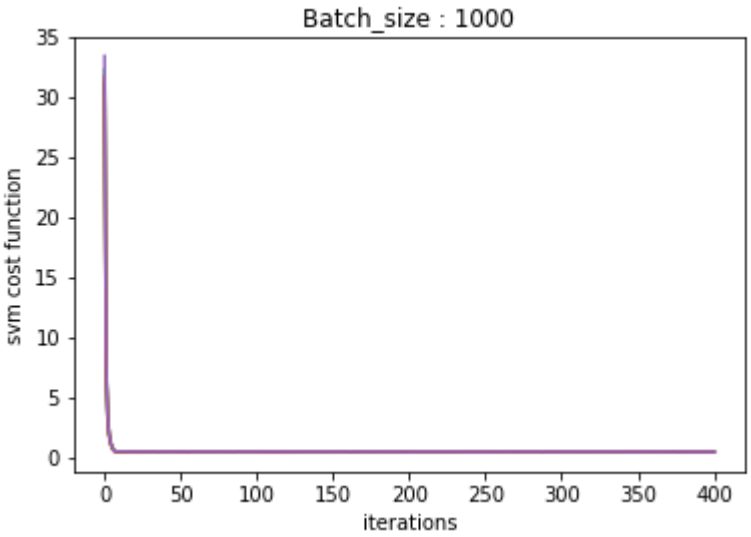
Batch Size (k) = 20
Number of runs = 5
Mean Runtime = 2.5996386051 seconds
Standard Deviation Runtime = 1.4536558864 seconds



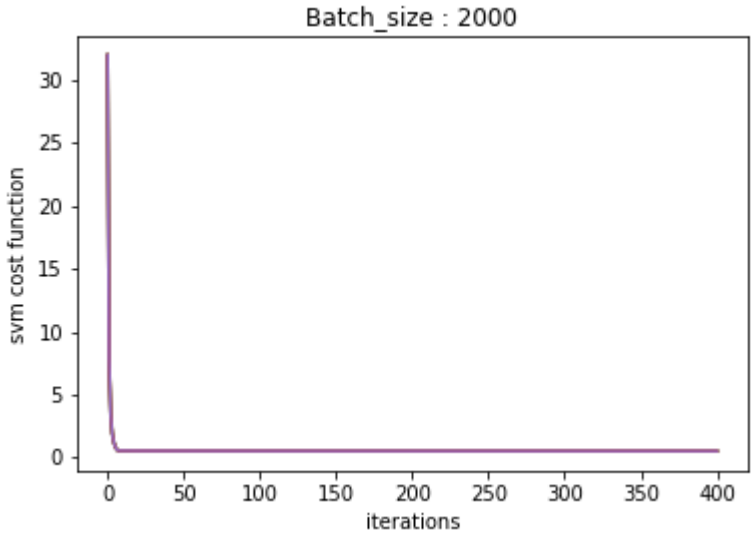
Batch Size (k) = 200
Number of runs = 5
Mean Runtime = 3.8312240600 seconds
Standard Deviation Runtime = 1.8121650722 seconds



Batch Size (k) = 1000
Number of runs = 5
Mean Runtime = 6.1666266918 seconds
Standard Deviation Runtime = 0.7572492165 seconds



Batch Size (k) = 2000
Number of runs = 5
Mean Runtime = 1.6600903987 seconds
Standard Deviation Runtime = 0.0935010046 seconds



b)

Derivation of the Gradient of the **Softplus** function is as follows,

$$\nabla L(w) = \frac{1}{N} \sum_{i=1:N} \nabla \left[a \log\left(1 + e^{\frac{1 - y_i w^T x_i}{a}}\right) + \lambda \|w\|^2 \right]$$

$$= \frac{1}{N} \sum_{i=1:N} \nabla \left[a \log\left(1 + e^{\frac{1 - y_i w^T x_i}{a}}\right) \right] + \nabla \left[\lambda \|w\|^2 \right]$$

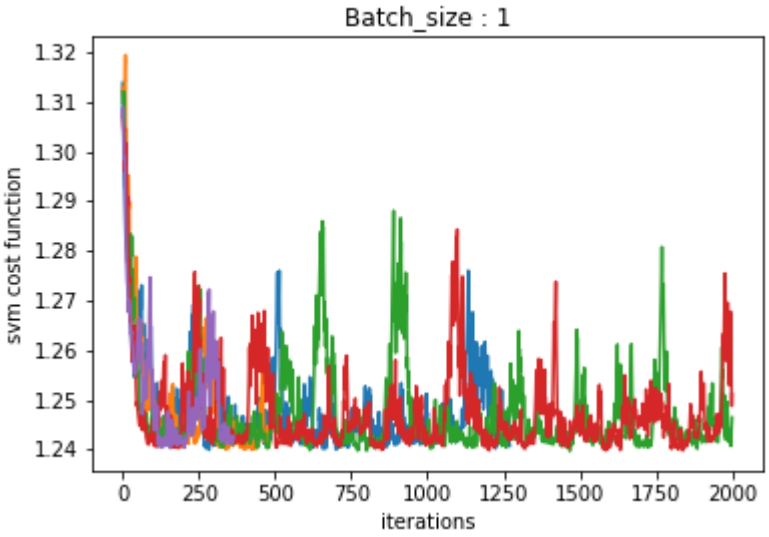
$$= \frac{1}{N} \sum_{i=1:N} \left[\left(\frac{a}{1 + e^{\frac{1 - y_i w^T x_i}{a}}} \times e^{\frac{1 - y_i w^T x_i}{a}} \times \left(\frac{1}{a}\right) \times (-x_i y_i) \right) + [2\lambda w] \right]$$

$$= \frac{1}{N} \sum_{i=1:N} \left[\frac{-x_i y_i}{1 + e^{\frac{y_i w^T x_i - 1}{a}}} + 2\lambda w \right]$$

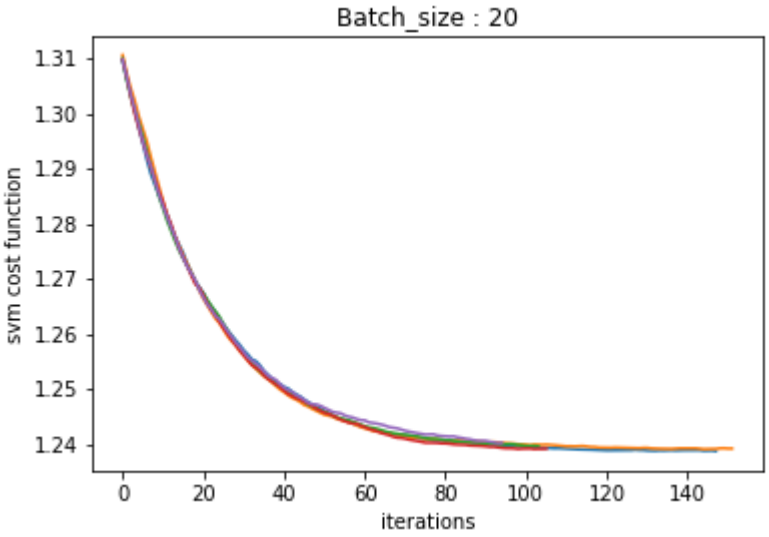
$$\nabla L(w) = \left(\frac{1}{N} \sum_{i=1:N} \frac{-x_i y_i}{1 + e^{\frac{y_i w^T x_i - 1}{a}}} \right) + 2\lambda w$$

The mean and standard deviation of runtimes we obtained using the **Softplus algorithm** are as follows for different k values.

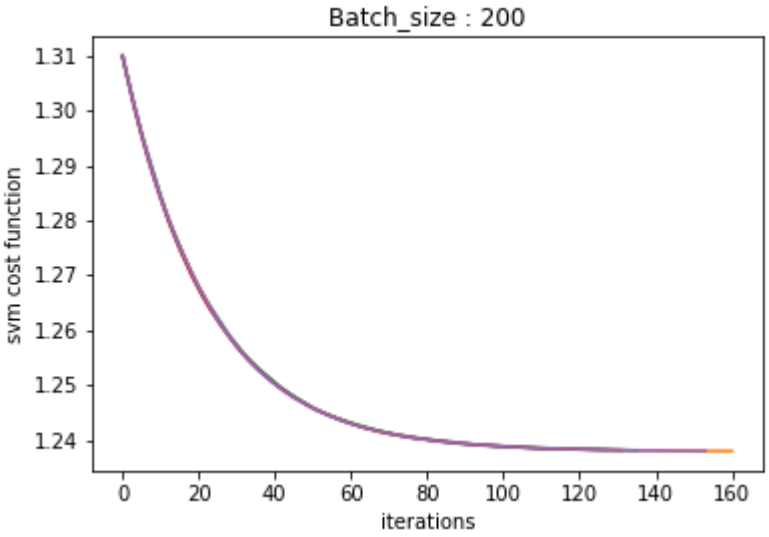
Batch Size (k) = 1
Number of runs = 5
Mean Runtime = 43.4661227703 seconds
Standard Deviation Runtime = 30.3375075746 seconds



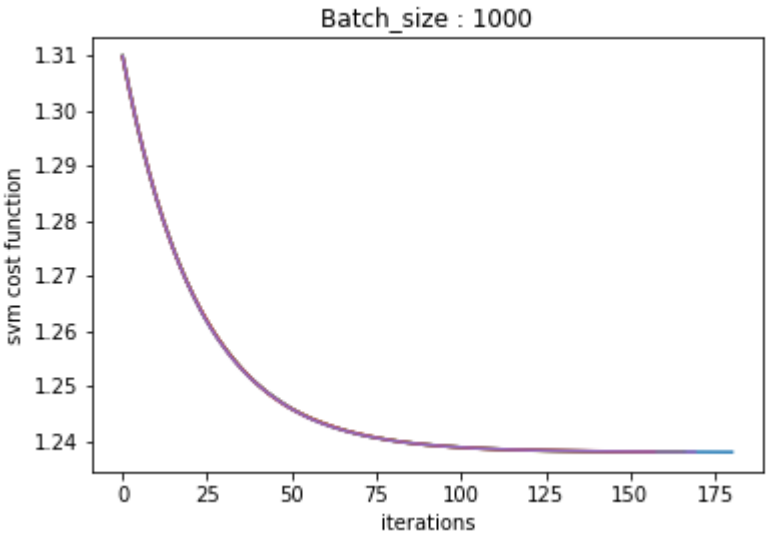
Batch Size (k) = 20
Number of runs = 5
Mean Runtime = 12.4473050117 seconds
Standard Deviation Runtime = 2.3757711437 seconds



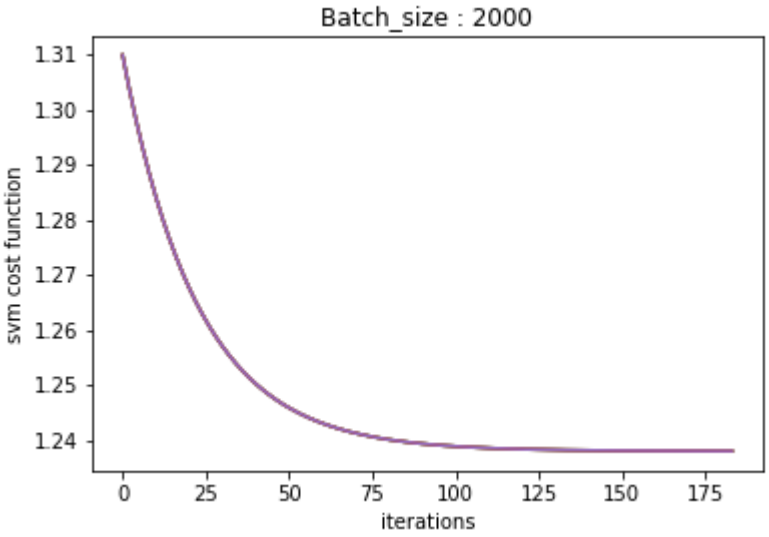
Batch Size (k) = 200
Number of runs = 5
Mean Runtime = 15.3314484596 seconds
Standard Deviation Runtime = 1.4831238377 seconds



Batch Size (k) = 1000
Number of runs = 5
Mean Runtime = 22.4613066673 seconds
Standard Deviation Runtime = 0.7229130157 seconds



Batch Size (k) = 2000
Number of runs = 5
Mean Runtime = 14.7364604949 seconds
Standard Deviation Runtime = 4.7252624717 seconds



The **Pegasos** algorithm performs stochastic gradient descent on a non-smooth function (hinge loss) whereas the other version uses a soft function variant of hinge loss called **Softplus**, we observed that the Cost function has a steeper curve or the cost value decreased faster (better convergence) in **Softplus** variant than in **Hinge loss** variant (Pegasos) as we know **stochastic gradient descent** reaches the optimum value much faster in smoother functions compared to non-smoother functions.