INLP Assignment - 3

Name :- MallaSailesh
Roll :- 2021101106

<u>SVD Based Word Embeddings</u>

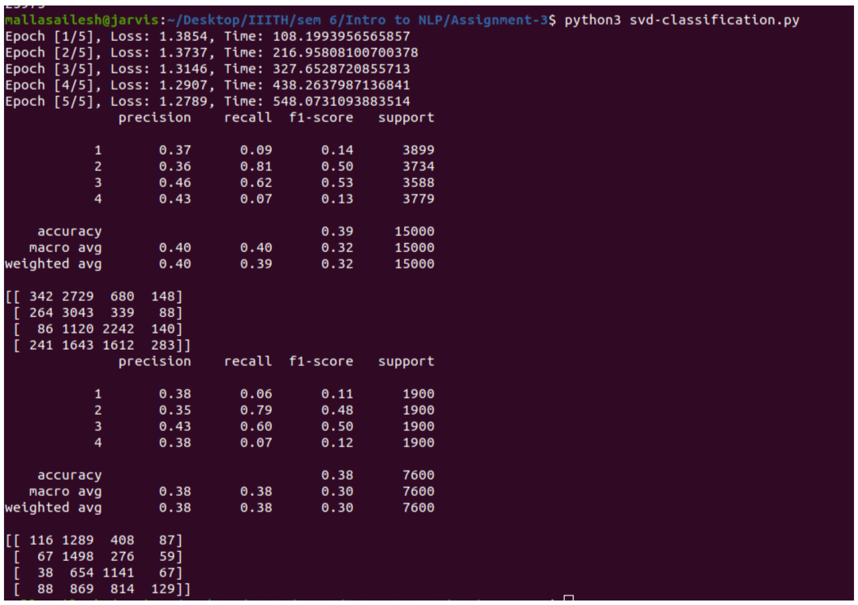
For SVD based word embeddings , the window Sizes which i took were :-

- 1
- 3
- 5 (because when i looked up google it said that 5 is a good window size when doing this kind of task even the glove model has window size 5). So this is the best I took.

Embedding length i took was - 300

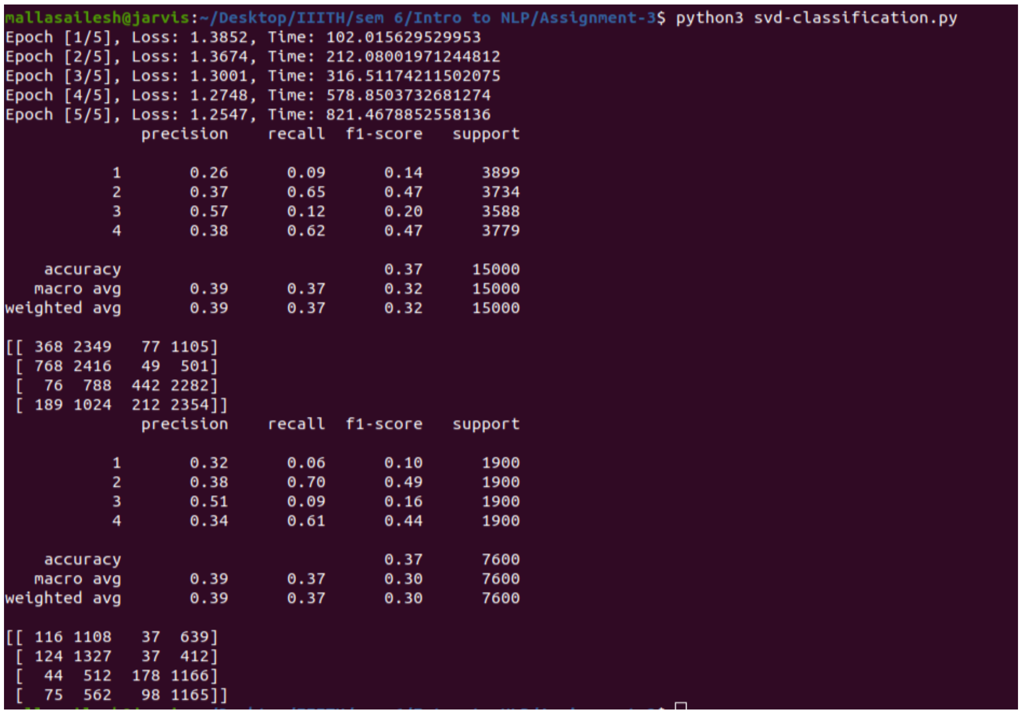
1, 3 I took for comparison purposes , I know with 1 less word distribution of a certain word is known, but i want to check how it works even though . And 3 to know how the values vary as we go.

For 1 :-



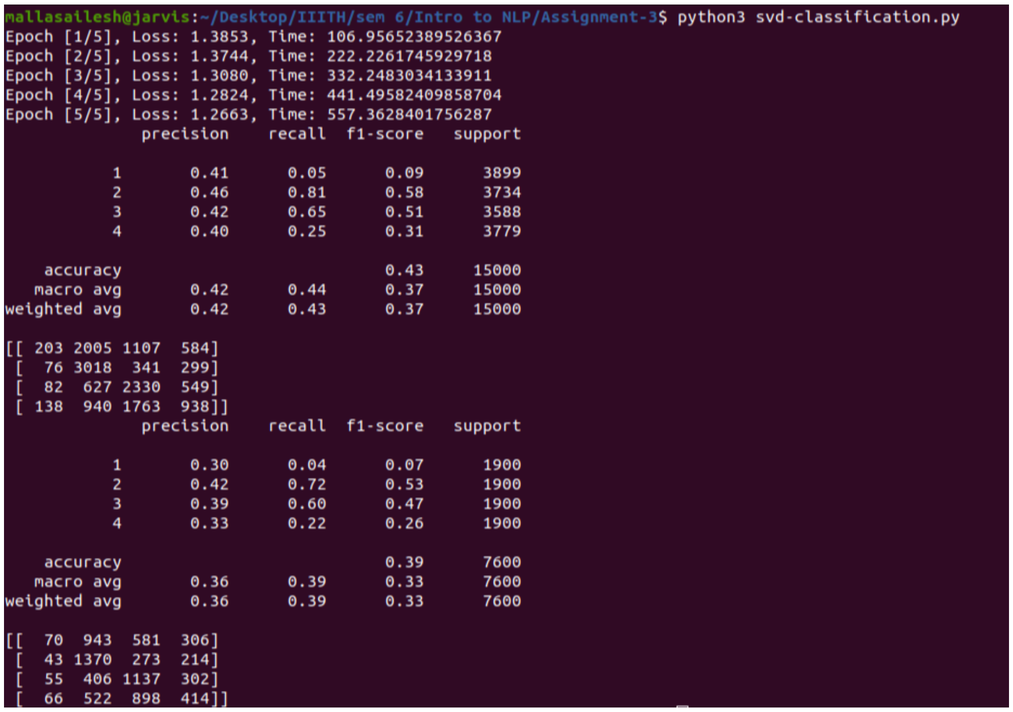
```
mallasailesh@jarvis:~/Desktop/IIITH/sem 6/Intro to NLP/Assignment-3$ python3 svd-classification.py
Epoch [1/5], Loss: 1.3854, Time: 108.1993956565857
Epoch [2/5], Loss: 1.3737, Time: 216.95808100700378
Epoch [3/5], Loss: 1.3146, Time: 327.6528720855713
Epoch [4/5], Loss: 1.2907, Time: 438.2637987136841
Epoch [5/5], Loss: 1.2789, Time: 548.0731093883514
              precision    recall  f1-score   support

           1       0.37      0.09      0.14      3899
           2       0.36      0.81      0.50      3734
           3       0.46      0.62      0.53      3588
           4       0.43      0.07      0.13      3779

    accuracy                           0.39     15000
   macro avg       0.40      0.40      0.32     15000
weighted avg       0.40      0.39      0.32     15000

[[ 342 2729  680  148]
 [ 264 3043  339   88]
 [  86 1120 2242  140]
 [ 241 1643 1612  283]]
              precision    recall  f1-score   support

           1       0.38      0.06      0.11      1900
           2       0.35      0.79      0.48      1900
           3       0.43      0.60      0.50      1900
           4       0.38      0.07      0.12      1900

    accuracy                           0.38      7600
   macro avg       0.38      0.38      0.30      7600
weighted avg       0.38      0.38      0.30      7600

[[ 116 1289  408   87]
 [  67 1498  276   59]
 [  38  654 1141   67]
 [  88  869  814  129]]
```

For 3:-



```
mallasailesh@jarvis:~/Desktop/IIITH/sem 6/Intro to NLP/Assignment-3$ python3 svd-classification.py
Epoch [1/5], Loss: 1.3852, Time: 102.015629529953
Epoch [2/5], Loss: 1.3674, Time: 212.08001971244812
Epoch [3/5], Loss: 1.3001, Time: 316.51174211502075
Epoch [4/5], Loss: 1.2748, Time: 578.8503732681274
Epoch [5/5], Loss: 1.2547, Time: 821.4678852558136
              precision    recall  f1-score   support

           1       0.26      0.09      0.14      3899
           2       0.37      0.65      0.47      3734
           3       0.57      0.12      0.20      3588
           4       0.38      0.62      0.47      3779

    accuracy                           0.37     15000
   macro avg       0.39      0.37      0.32     15000
weighted avg       0.39      0.37      0.32     15000

[[ 368 2349   77 1105]
 [ 768 2416   49  501]
 [  76  788  442 2282]
 [ 189 1024  212 2354]]
              precision    recall  f1-score   support

           1       0.32      0.06      0.10      1900
           2       0.38      0.70      0.49      1900
           3       0.51      0.09      0.16      1900
           4       0.34      0.61      0.44      1900

    accuracy                           0.37      7600
   macro avg       0.39      0.37      0.30      7600
weighted avg       0.39      0.37      0.30      7600

[[ 116 1108   37  639]
 [ 124 1327   37  412]
 [  44  512  178 1166]
 [  75  562   98 1165]]
```

For 5:-



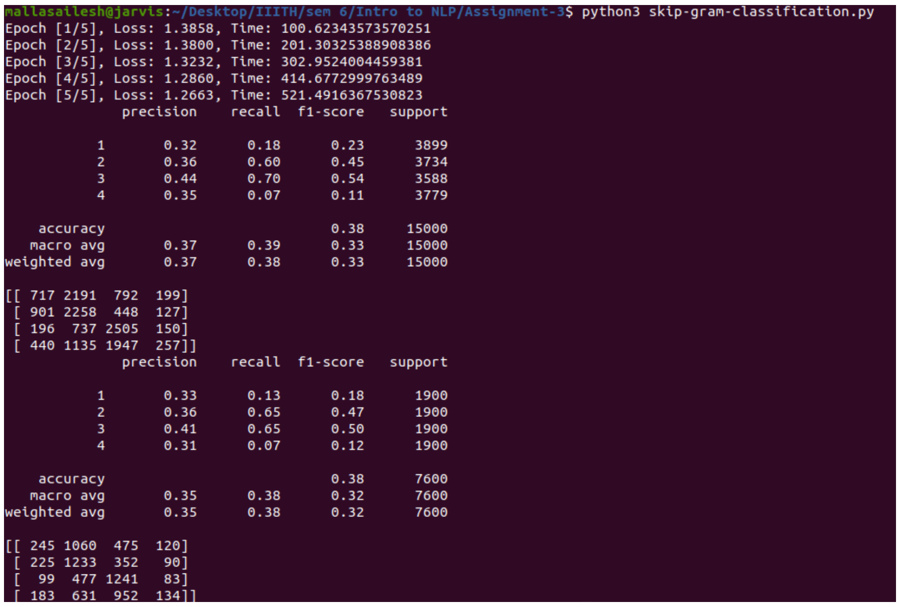
```
mallasailesh@jarvis:~/Desktop/IIITH/sem 6/Intro to NLP/Assignment-3$ python3 svd-classification.py
Epoch [1/5], Loss: 1.3853, Time: 106.95652389526367
Epoch [2/5], Loss: 1.3744, Time: 222.2261745929718
Epoch [3/5], Loss: 1.3080, Time: 332.2483034133911
Epoch [4/5], Loss: 1.2824, Time: 441.49582409858704
Epoch [5/5], Loss: 1.2663, Time: 557.3628401756287
              precision    recall  f1-score   support

           1       0.41      0.05      0.09      3899
           2       0.46      0.81      0.58      3734
           3       0.42      0.65      0.51      3588
           4       0.40      0.25      0.31      3779

    accuracy                           0.43     15000
   macro avg       0.42      0.44      0.37     15000
weighted avg       0.42      0.43      0.37     15000

[[ 203 2005 1107  584]
 [  76 3018  341  299]
 [  82  627 2330  549]
 [ 138  940 1763  938]]
              precision    recall  f1-score   support

           1       0.30      0.04      0.07      1900
           2       0.42      0.72      0.53      1900
           3       0.39      0.60      0.47      1900
           4       0.33      0.22      0.26      1900

    accuracy                           0.39      7600
   macro avg       0.36      0.39      0.33      7600
weighted avg       0.36      0.39      0.33      7600

[[  70  943  581  306]
 [  43 1370  273  214]
 [  55  406 1137  302]
 [  66  522  898  414]]
```

Skip-Gram with Negative Sampling

For the same above reasons the window sizes were :-

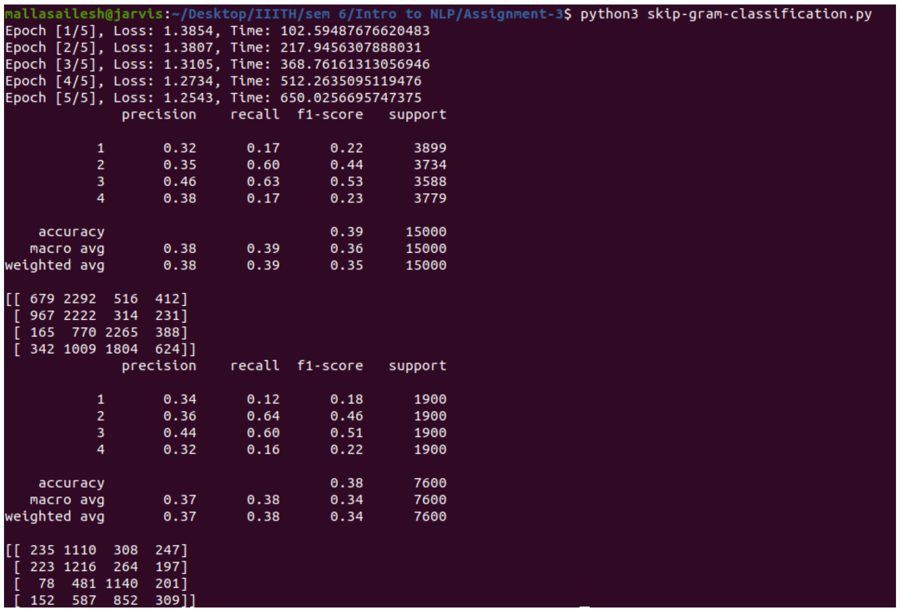
- 1
- 3
- 5

For 1 :-



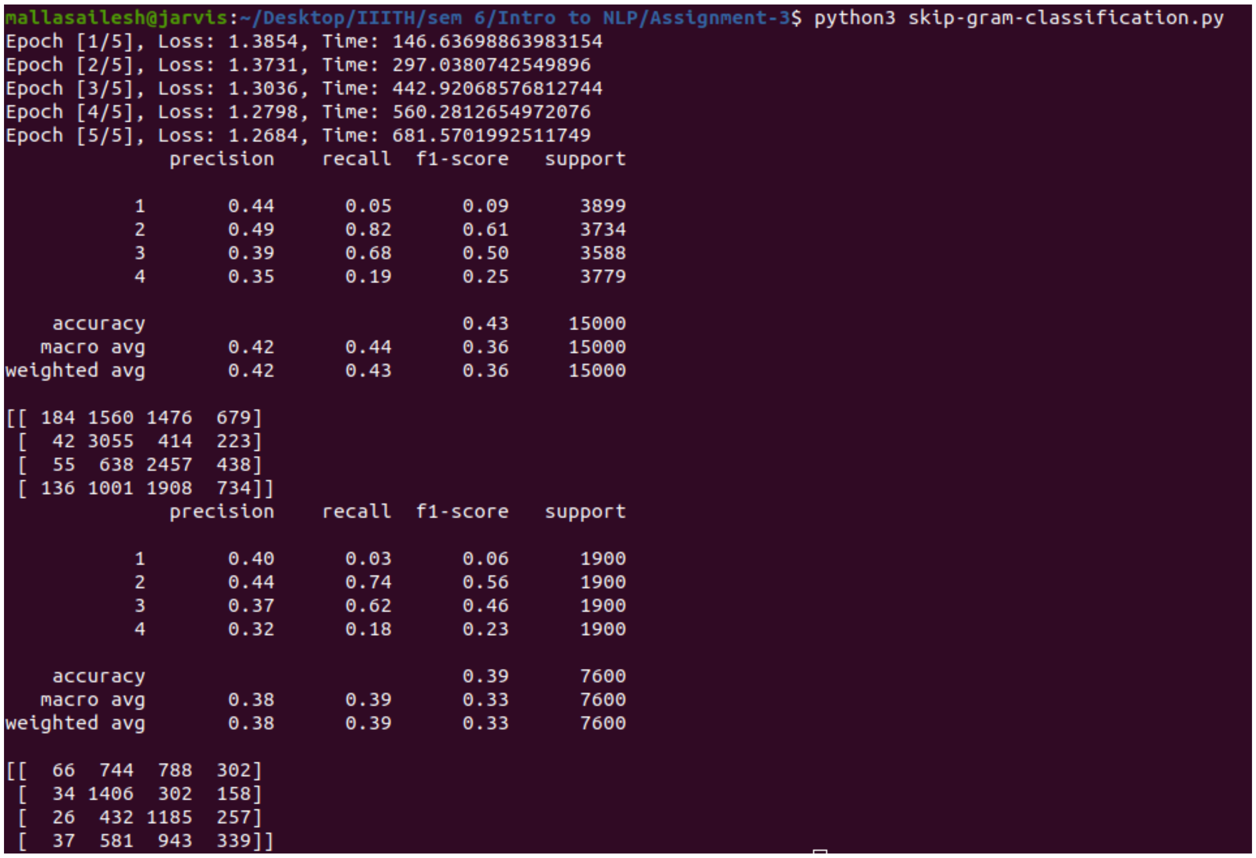
```
mallasailesh@jarvis:~/Desktop/IIITH/sem 6/Intro to NLP/Assignment-3$ python3 skip-gram-classification.py
Epoch [1/5], Loss: 1.3858, Time: 100.62343573570251
Epoch [2/5], Loss: 1.3800, Time: 201.30325388908386
Epoch [3/5], Loss: 1.3232, Time: 302.9524004459381
Epoch [4/5], Loss: 1.2860, Time: 414.6772999763489
Epoch [5/5], Loss: 1.2663, Time: 521.4916367530823
              precision    recall  f1-score   support

           1       0.32      0.18      0.23      3899
           2       0.36      0.60      0.45      3734
           3       0.44      0.70      0.54      3588
           4       0.35      0.07      0.11      3779

    accuracy                           0.38     15000
   macro avg       0.37      0.39      0.33     15000
weighted avg       0.37      0.38      0.33     15000

[[ 717 2191  792  199]
 [ 901 2258  448  127]
 [ 196  737 2505  150]
 [ 440 1135 1947  257]]
              precision    recall  f1-score   support

           1       0.33      0.13      0.18      1900
           2       0.36      0.65      0.47      1900
           3       0.41      0.65      0.50      1900
           4       0.31      0.07      0.12      1900

    accuracy                           0.38      7600
   macro avg       0.35      0.38      0.32      7600
weighted avg       0.35      0.38      0.32      7600

[[ 245 1060  475  120]
 [ 225 1233  352   90]
 [  99  477 1241   83]
 [ 183  631  952  134]]
```

For 3:-



```
mallasailesh@jarvis:~/Desktop/IIITH/sem 6/Intro to NLP/Assignment-3$ python3 skip-gram-classification.py
Epoch [1/5], Loss: 1.3854, Time: 102.59487676620483
Epoch [2/5], Loss: 1.3807, Time: 217.9456307888031
Epoch [3/5], Loss: 1.3105, Time: 368.76161313056946
Epoch [4/5], Loss: 1.2734, Time: 512.2635095119476
Epoch [5/5], Loss: 1.2543, Time: 650.0256695747375
              precision    recall  f1-score   support

           1       0.32      0.17      0.22      3899
           2       0.35      0.60      0.44      3734
           3       0.46      0.63      0.53      3588
           4       0.38      0.17      0.23      3779

    accuracy                           0.39     15000
   macro avg       0.38      0.39      0.36     15000
weighted avg       0.38      0.39      0.35     15000

[[ 679 2292  516  412]
 [ 967 2222  314  231]
 [ 165  770 2265  388]
 [ 342 1009 1804  624]]
              precision    recall  f1-score   support

           1       0.34      0.12      0.18      1900
           2       0.36      0.64      0.46      1900
           3       0.44      0.60      0.51      1900
           4       0.32      0.16      0.22      1900

    accuracy                           0.38      7600
   macro avg       0.37      0.38      0.34      7600
weighted avg       0.37      0.38      0.34      7600

[[ 235 1110  308  247]
 [ 223 1216  264  197]
 [  78  481 1140  201]
 [ 152  587  852  309]]
```

For 5:-



```
mallasailesh@jarvis:~/Desktop/IIITH/sem 6/Intro to NLP/Assignment-3$ python3 skip-gram-classification.py
Epoch [1/5], Loss: 1.3854, Time: 146.63698863983154
Epoch [2/5], Loss: 1.3731, Time: 297.0380742549896
Epoch [3/5], Loss: 1.3036, Time: 442.92068576812744
Epoch [4/5], Loss: 1.2798, Time: 560.2812654972076
Epoch [5/5], Loss: 1.2684, Time: 681.5701992511749
              precision    recall  f1-score   support

           1       0.44      0.05      0.09      3899
           2       0.49      0.82      0.61      3734
           3       0.39      0.68      0.50      3588
           4       0.35      0.19      0.25      3779

    accuracy                           0.43     15000
   macro avg       0.42      0.44      0.36     15000
weighted avg       0.42      0.43      0.36     15000

[[ 184 1560 1476  679]
 [  42 3055  414  223]
 [  55  638 2457  438]
 [ 136 1001 1908  734]]
              precision    recall  f1-score   support

           1       0.40      0.03      0.06      1900
           2       0.44      0.74      0.56      1900
           3       0.37      0.62      0.46      1900
           4       0.32      0.18      0.23      1900

    accuracy                           0.39      7600
   macro avg       0.38      0.39      0.33      7600
weighted avg       0.38      0.39      0.33      7600

[[  66  744  788  302]
 [  34 1406  302  158]
 [  26  432 1185  257]
 [  37  581  943  339]]
```

Possible Reasons for Performance Differences:

- Contextual Information: SGNS (Skip-Gram with Negative Sampling) captures local context better than SVD, which may lead to better performance in tasks that require understanding of word semantics.
- Dimensionality Reduction: SVD might struggle to capture nuanced semantic relationships due to its dimensionality reduction, while SGNS can learn more complex relationships directly from the training data.
- Training Data Size: If the training data is small, SVD may not be able to generalize well, whereas SGNS can learn better representations from limited data due to its ability to capture local context efficiently.

Shortcomings:

SVD Shortcomings:

- SVD may struggle with large datasets due to computational complexity.
- It may not capture subtle semantic relationships effectively.
- SVD requires large amounts of memory for storing the decomposed matrices.

SGNS Shortcomings:

- SGNS requires a large corpus for training to produce high-quality embeddings.
- It may not perform well on rare words or out-of-vocabulary terms.
- The quality of SGNS embeddings can degrade with insufficient negative sampling.

In conclusion, SGNS is likely to outperform SVD in tasks requiring understanding of semantic relationships due to its ability to capture local context efficiently. However, the choice between these methods should consider factors such as dataset size, computational resources, and the specific requirements of the task at hand.