Answer to Q1.1

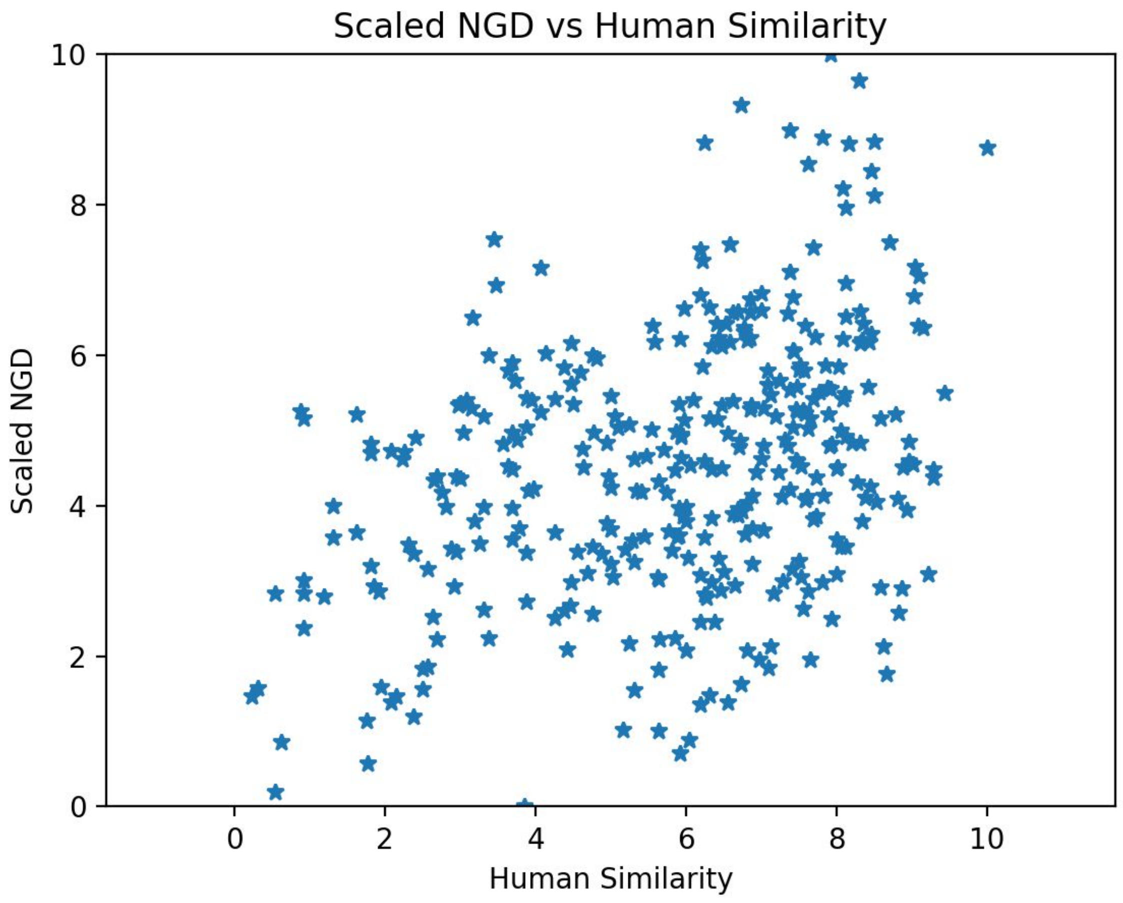
- To calculate **NGD** below formula was used:

$$\mathrm{NGD}(x,y) = \frac{\max\{\log f(x), \log f(y)\} - \log f(x,y)}{\log N - \min\{\log f(x), \log f(y)\}}$$

- f(x) = number of search results for word x (say ‘tiger’) and f(y) = number search results for word y (say ‘love’)
- f(x, y) is taken to be the number of search results for string x + y i.e. ‘tiger love’
- N is the total number of pages which is taken to be equal to the number of search results for word ‘a’ multiplied by 1000 (reference Wikipedia) which comes out to be, N = 25270000000000
- Google Search was done so as to display from 2nd results which helps reduce HTML content. So following string is scrapped from the searched page "Page 2 of about 25,27,00,00,000 results (0.38 seconds)”
- So 17 index of the string starts the count of no of pages found. So count was done till next space is found which gives us the number of search results.
- Google Search functioning is implemented in **googlesearch.py** rest as per the problem 1 is done in **q1.py**

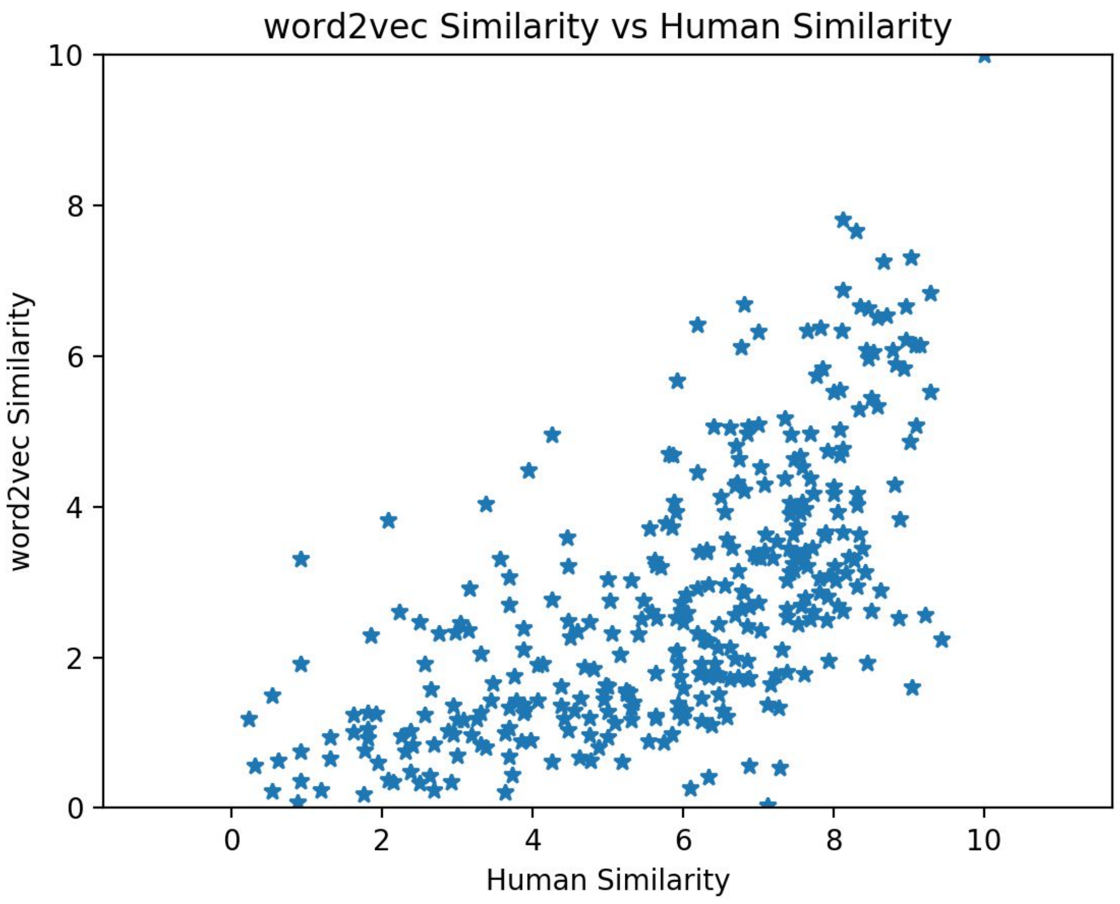
Answer to Q1.2

- A quite similar graph is obtained as present in the lecture slides with some variations, basically due to Google search query doesn't always gives similar results
- Scaling was done as below:
  $NGD_Scaled = max(NGD) - NGD$
  $NGD_Scaled = 10 * (NGD_Scaled / max(NGD_Scaled))$
- This scaling was done so as to get the data from 0 to 10 and also to make higher values to represent more similarity.



Answer to Q1.3

- As like 1.2 this was also done to produce the graph.
- Function to calculate similarity between two words uses GoogleNews-vectors-negative300 pretrained model.
- This graph was more closed i.e. points have less spread as compared to the NGD one.


word2vec Similarity vs Human Similarity
word2vec Similarity
Human Similarity