Question 5 answer :

show_nearest() outputs :

1) california
[('connecticut', 0.77585596670221912), ('florida', 0.73291716503442161), ('pennsylvania', 0.72574956490099052), ('texas', 0.72516872445749747), ('massachusetts', 0.71518805065916291), ('ohio', 0.69811989853498302), ('virginia', 0.68497574631807945), ('southern', 0.58758906600720351), ('westchester', 0.55159949690771992), ('state', 0.5434206801909317)]

Comment: This makes some sense than our previous version as California is treated in state context for eg. It is connected with other states in United States rather than getting similarity with words like 'france'.
This method definitely produces better results than the previous one.

2) doctors
[('patients', 0.84326514940394681), ('researchers', 0.69631288531223068), ('hospitals', 0.67635524220387744), ('scientists', 0.65953150789015402), ('parents', 0.63919972266239944), ('drugs', 0.63584921439379583), ('treatment', 0.63476112181020017), ('detainees', 0.62771508048987079), ('workers', 0.6177949524207943), ('employees', 0.6148638317748647)]

Comment: This result shown is making more sense than our previous result as word 'doctors' is usually taken in a context where we use 'patients' & 'hospitals'. Previous implementation treated word in a context of profession.
This method definitely produces better results than the previous one for above keyword.

3) small
[('large', 0.87211660144814551), ('tiny', 0.74083589489471058), ('vast', 0.66404896062719665), ('huge', 0.64144538065430368), ('smaller', 0.62626460389303096), ('big', 0.58646670179017757), ('larger', 0.58372874406169295), ('separate', 0.55406440213227459), ('massive', 0.5502630495530989), ('wide', 0.51430426697901355)]

Comment: In this method also we are getting almost similar result for 'small' word and which makes sense as these highly co-used words. One difference we can see is that words like 'tiny' and 'huge' have high similarity score and words like 'smaller' , 'larger' found place in top 10 similar words.
Hence ,This method produces better results than the previous one for above keyword.

4) draw
[('bring', 0.7446555816252256), ('carry', 0.70349063378111765), ('reach', 0.69466568535683648), ('give', 0.68667783445894692), ('hold', 0.68417836291900469), ('turn', 0.67374496227443803), ('get', 0.66979874601866118), ('move', 0.66592329053695432), ('pull', 0.65965792986231486), ('gain', 0.6548408899521273)]

Comment: This is a perfect example where this method is producing far better result than our previous implementation. Our word 'draw' is used mostly in similar meaning as to 'pull' or 'get'. These words found place in top 10 similar words. Previous method

was producing some non meaningful result fir this keyword.
Hence, This method produces better results than the previous one for above keyword.

5) month
[('week', 0.94751461446839091), ('year', 0.92575855145514208), ('decade', 0.72238092650921748), ('november', 0.71491260764834541), ('weekend', 0.68522150863191689), ('fall', 0.68376272045645448), ('semester', 0.67363839870683473), ('summer', 0.66832973277951324), ('october', 0.66104304618165843), ('september', 0.65245555995719684)]

Comment: This makes sense and is almost similar to the previous method in terms of results by providing some more better words like 'october', 'september'and 'november' which are actual months names.

6) france
[('germany', 0.88629873133296277), ('japan', 0.82074796018738605), ('britain', 0.81791720915691435), ('spain', 0.81499170449648695), ('italy', 0.81353408919847925), ('brazil', 0.78883649810722012), ('europe', 0.77679133432263714), ('argentina', 0.77324558044728509), ('india', 0.77284882639929242), ('russia', 0.77120234172741864)]

Comment: This makes sense and is almost similar to the previous method in terms of result output.

In general this **approach** (cosine dense) produces far better results than our previous approach (cosine sparse) and produces more context aware similar words. As cosine dense method uses word embeddings as input to similarity measures, it provides better result than cosine sparse.