# Latent Semantic Indexing and Word Vector Representations

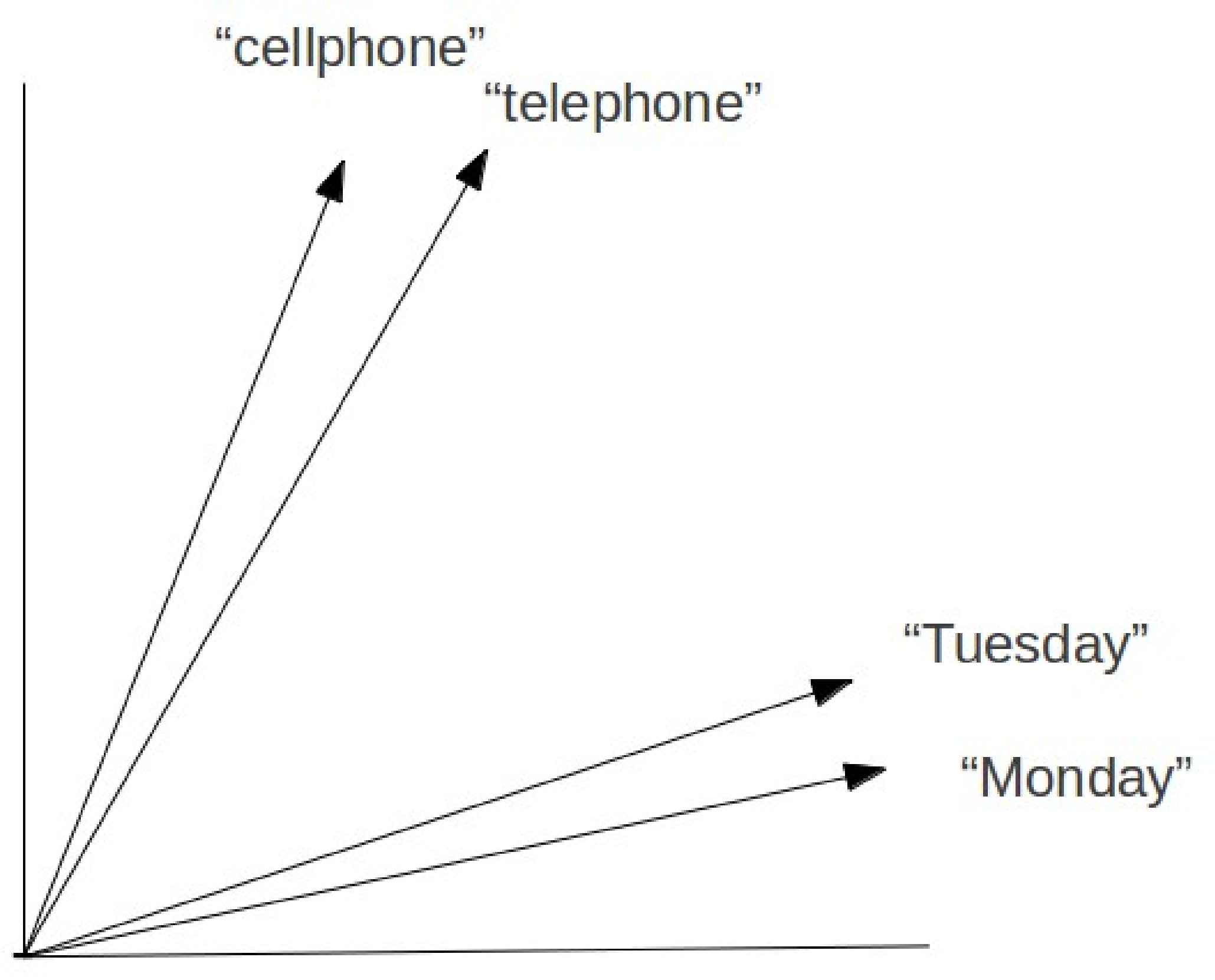
Trevor Barron
Adviser: Matthew Whitehead – Colorado College

## Introduction

- In a probabilistic *N*-gram natural language model each word is represented as a single element. A word's representation does not distinguish it from any other word and all words are equally related. “Monday” and “telephone” are just as related as “Monday” and “Tuesday” in this model.
- Representing words using semantic feature vectors in an *N* dimensional space permits comparison of words by the distance between their respective vectors. The distance between the vectors representing “Monday” and “Tuesday” is much less than the distance between the vectors for “Monday” and “telephone.”
- One can find a vector representation for a word by latent semantic indexing (LSI) on a large text corpus.
- This research explores using neural networks to predict the abstract *meaning* of future words during text generation using vector representations for words.

## Objectives

Perform LSI on a large text corpus to generate a vector representation for words in *N* dimensional space. Use these vectors as input and output to train a neural network to understand the mapping between and abstract input and an abstract output.



## Methods

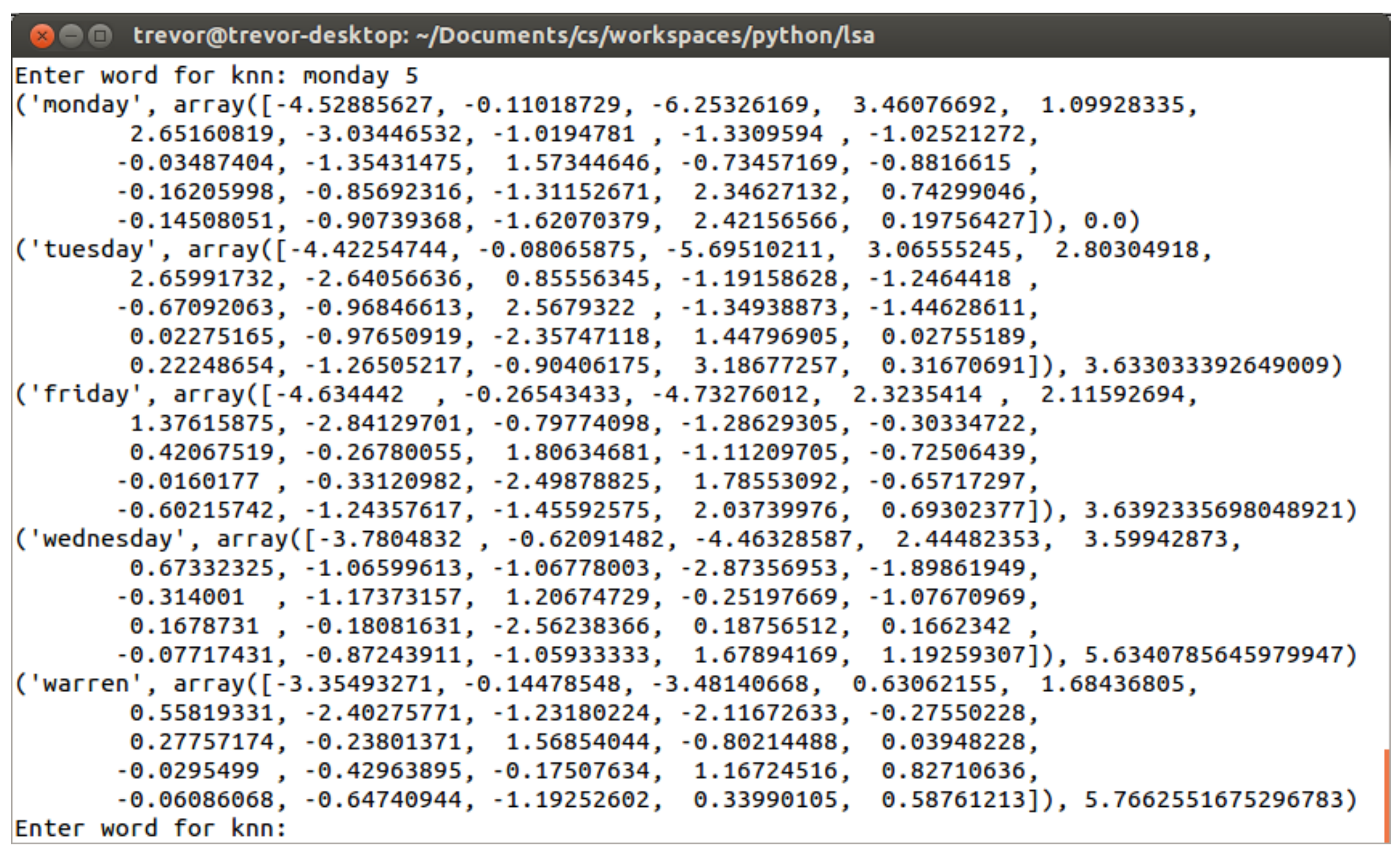
- LSI term document matrix X
  - X(*i, j*) = Count(term *i* in document *j*)
  - Term frequency – inverse document frequency preconditioning of *X* resulted in significant improvements.
- Use Singular Value Decomposition (SVD) to find an approximate vector representation

$$\overset{\overset{X}{(\mathbf{d}_j)\downarrow}}{}(\mathbf{t}_i^T) \rightarrow \begin{bmatrix} x_{1,1} & \dots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{m,1} & \dots & x_{m,n} \end{bmatrix} = (\hat{\mathbf{t}}_i^T) \rightarrow \overset{U}{\begin{bmatrix} \begin{bmatrix} \mathbf{u}_1 \end{bmatrix} \dots \begin{bmatrix} \mathbf{u}_l \end{bmatrix} \end{bmatrix}} \cdot \overset{\Sigma}{\begin{bmatrix} \sigma_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_l \end{bmatrix}} \cdot \overset{\overset{V^T}{(\hat{\mathbf{d}}_j)\downarrow}}{\begin{bmatrix} [ & \mathbf{v}_1 & ] \\ & \vdots & \\ [ & \mathbf{v}_l & ] \end{bmatrix}}$$

- Find the k-rank reduction by finding the product of the *k* largest values in Σ and the associated vectors in *U*.
  - The vector for term *ti* = *U*Σ[*i*], the *i*-th row of *U*Σ.
- Use the vectors of dataset trigrams as input and output to train a neural network
  - Example: “the morning sun”
    - the = [-109.2, 5.9, 4.4, 3.8, 1.0]; morning = [-10.1, -8.6, -0.6, 1.6, -0.9]; sun = [-6.9, -4.7, 2.9, 1.6, 0.6]
    - Input = [-109.2, 5.9, 4.4, 3.8, 1.0, -10.1, -8.6, -0.6, 1.6, -0.9]
    - Output = [-6.9, -4.7, 2.9, 1.6, 0.6]

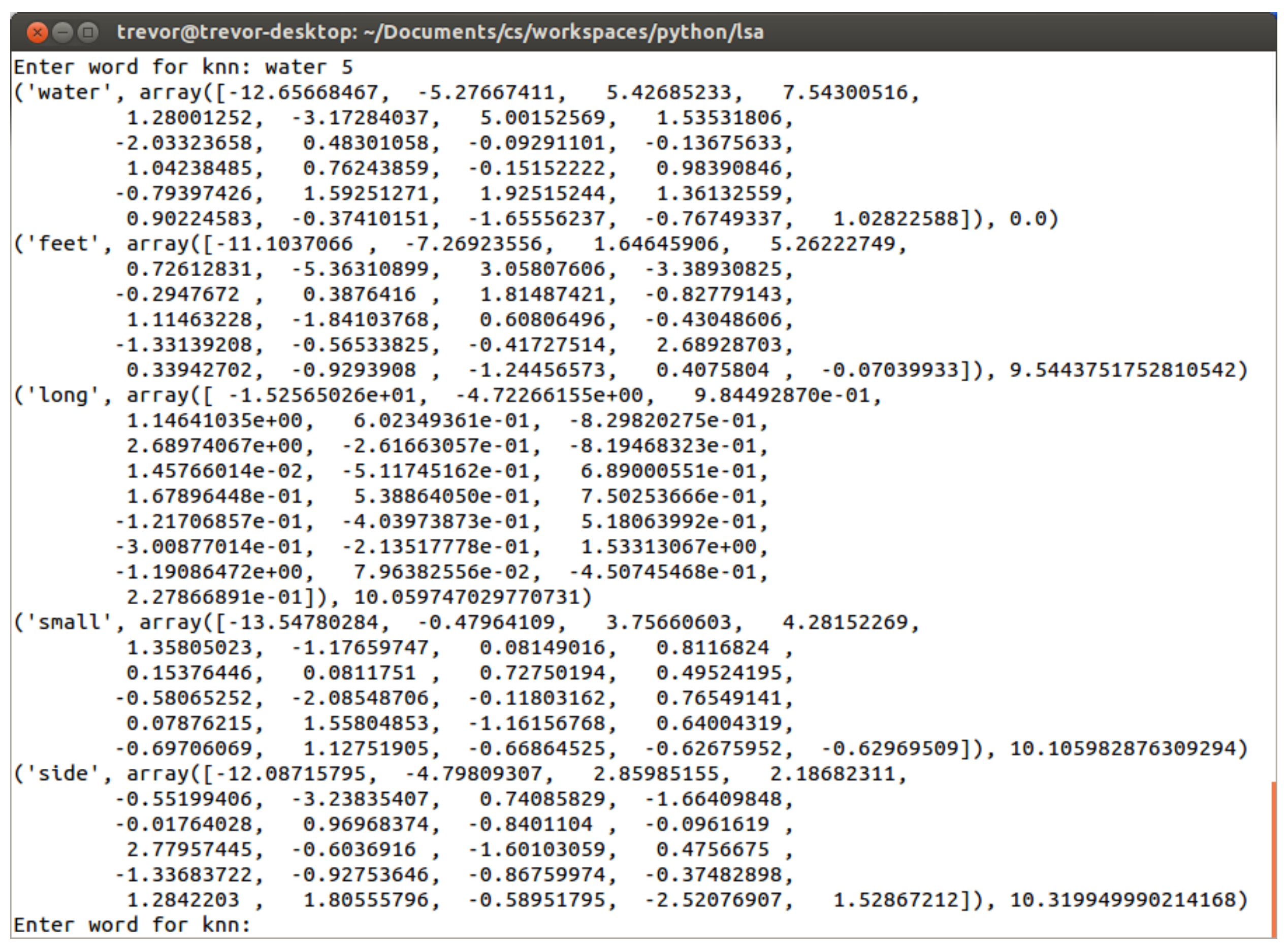
## Results

- Using the Brown corpus (500 documents and 46758 unique terms after preprocessing) to generate vector representations.
  - Sample KNN:



```
trevor@trevor-desktop: ~/Documents/cs/workspaces/python/lsa
Enter word for knn: monday 5
('monday', array([-4.52885627, -0.11018729, -6.25326169,  3.46076692,  1.09928335,
        2.65160819, -3.03446532, -1.0194781 , -1.3309594 , -1.02521272,
       -0.03487404, -1.35431475,  1.57344646, -0.73457169, -0.8816615 ,
       -0.16205998, -0.85692316, -1.31152671,  2.34627132,  0.74299046,
       -0.14508051, -0.90739368, -1.62070379,  2.42156566,  0.19756427]), 0.0)
('tuesday', array([-4.42254744, -0.08065875, -5.69510211,  3.06555245,  2.80304918,
        2.65991732, -2.64056636,  0.85556345, -1.19158628, -1.2464418 ,
       -0.67092063, -0.96846613,  2.5679322 , -1.34938873, -1.44628611,
        0.02275165, -0.97650919, -2.35747118,  1.44796905,  0.02755189,
        0.22248654, -1.26505217, -0.90406175,  3.18677257,  0.31670691]), 3.633033392649009)
('friday', array([-4.634442  , -0.26543433, -4.73276012,  2.3235414 ,  2.11592694,
        1.37615875, -2.84129701, -0.79774098, -1.28629305, -0.30334722,
        0.42067519, -0.26780055,  1.80634681, -1.11209705, -0.72506439,
       -0.0160177 , -0.33120982, -2.49878825,  1.78553092, -0.65717297,
       -0.60215742, -1.24357617, -1.45592575,  2.03739976,  0.69302377]), 3.639233569804892l)
('wednesday', array([-3.7804832 , -0.62091482, -4.46328587,  2.44482353,  3.59942873,
        0.67332325, -1.06599613, -1.06778003, -2.87356953, -1.89861949,
       -0.314001  , -1.17373157,  1.20674729, -0.25197669, -1.07670969,
        0.1678731 , -0.18081631, -2.56238366,  0.18756512,  0.1662342 ,
       -0.07717431, -0.87243911, -1.05933333,  1.67894169,  1.19259307]), 5.634078564597947)
('warren', array([-3.35493271, -0.14478548, -3.48140668,  0.63062155,  1.68436805,
        0.55819331, -2.40275771, -1.23180224, -2.11672633, -0.27550228,
        0.27757174, -0.23801371,  1.56854044, -0.80214488,  0.03948228,
       -0.0295499 , -0.42963895, -0.17507634,  1.16724516,  0.82710636,
       -0.06086068, -0.64740944, -1.19252602,  0.33990105,  0.58761213]), 5.766255167529678З)
Enter word for knn:
```

- This works very well for some cases as seen above but very poorly for others. A larger dataset may resolve some of these problems.



```
trevor@trevor-desktop: ~/Documents/cs/workspaces/python/lsa
Enter word for knn: water 5
('water', array([-12.65668467,  -5.27667411,   5.42685233,   7.54300516,
         1.28001252,  -3.17284037,   5.00152569,   1.53531806,
        -2.03323658,   0.48301058,  -0.09291101,  -0.13675633,
         1.04238485,   0.76243859,  -0.15152222,   0.98390846,
        -0.79397426,   1.59251271,   1.92515244,   1.36132559,
         0.90224583,  -0.37410151,  -1.65556237,  -0.76749337,   1.02822588]), 0.0)
('feet', array([-11.1037066 ,  -7.26923556,   1.64645906,   5.26222749,
         0.72612831,  -5.36310899,   3.05807606,  -3.38930825,
        -0.2947672 ,   0.3876416 ,   1.81487421,  -0.82779143,
         1.11463228,  -1.84103768,   0.60806496,  -0.43048606,
        -1.33139208,  -0.56533825,  -0.41727514,   2.68928703,
         0.33942702,  -0.9293908 ,  -1.24456573,   0.4075804 ,  -0.07039933]), 9.544375175281054 2)
('long', array([ -1.52565026e+01,  -4.72266155e+00,   9.84492870e-01,
         1.14641035e+00,   6.02349361e-01,  -8.29820275e-01,
         2.68974067e+00,  -2.61663057e-01,  -8.19468323e-01,
         1.45766014e-02,  -5.11745162e-01,   6.89000551e-01,
         1.67896448e-01,   5.38864050e-01,   7.50253666e-01,
        -1.21706857e-01,  -4.03973873e-01,   5.18063992e-01,
        -3.00877014e-01,  -2.13517778e-01,   1.53313067e+00,
        -1.19086472e+00,   7.96382556e-02,  -4.50745468e-01,
         2.27866891e-01]), 10.059747029770731)
('small', array([-13.54780284,  -0.47964109,   3.75660603,   4.28152269,
         1.35805023,  -1.17659747,   0.08149016,   0.8116824 ,
         0.15376446,   0.0811751 ,   0.72750194,   0.49524195,
        -0.58065252,  -2.08548706,  -0.11803162,   0.76549141,
         0.07876215,   1.55804853,  -1.16156768,   0.64004319,
        -0.69706069,   1.12751905,  -0.66864525,  -0.62675952,  -0.62969509]), 10.105982876309294)
('side', array([-12.08715795,  -4.79809307,   2.85985155,   2.18682311,
        -0.55199406,  -3.23835407,   0.74085829,  -1.66409848,
        -0.01764028,   0.96968374,  -0.8401104 ,  -0.0961619 ,
         2.77957445,  -0.6036916 ,  -1.60103059,   0.4756675 ,
        -1.33683722,  -0.92753646,  -0.86759974,  -0.37482898,
         1.2842203 ,   1.80555796,  -0.58951795,  -2.52076907,   1.52867212]), 10.319949990214168)
Enter word for knn:
```

- Neural network learning was difficult even for carefully chosen datasets.

## Conclusion

- It takes a lot of effort to find a good vector representation
- SVD is quite sensitive to the condition of the matrix. One very common word like “the” can throw off the entire set of vectors
- In almost every case, more data equals a better result.
- A data formatting problem may be causing the neural network training difficulties. Moving forward one might ensure that the input & output are produced as expected and perhaps try alternative methods of normalization.

## References

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, Richard Harshman (1990). "Indexing by Latent Semantic Analysis" (PDF). Journal of the American Society for Information Science 41 (6): 391–407.

W. N. Francis, H. Kucera, A Standard Corpus of Present-Day Edited American English, for use with Digital Computers (Brown Corpus). 1964.