The Curse of Dimensionality: Inside Out

Naveen Venkat[1]
Final Year Undergraduate Student
Dept. of CSIS, BITS Pilani

Abstract

The Curse of Dimensionality, introduced by Bellman [1], refers to the explosive nature of spatial dimensions and its resulting effects, such as, an exponential increase in computational effort, large waste of space and poor visualization capabilities. Higher number of dimensions theoretically allow more information to be stored, but practically rarely help due to the higher possibility of noise and redundancy in real world data. In this article, the effects of high dimensionality is studied through various experiments and the possible solutions to counter or mitigate such effects are proposed. The source code of the experiments performed is available publicly on github[2].

I. Introduction

In Machine Learning problems, an increase in the dimensionality of data results in the sparsification of data. Simply put, a small increase in the dimensionality would require a large increase in the volume of the data to maintain similar level of performance in tasks such as clustering, regression etc. Intuitively, we can view the explosive nature of spatial dimensions with a simple measure such as volume. Consider hypercubes with **0**, **1**, and **2** dimensions, each measuring **r** units. A hypercube with dimension **0** is just a point. Thus it has no volume. A hypercube with **1** dimension is a line, with the hypervolume defined by its length (**r**). In **2** dimensions, a hypercube is a square, with a hypervolume equal to its area ($\mathbf{r^2}$). Note, here the volume means the Euclidean volume. In general, a hypercube with **d** (> 0) dimensions has a volume given by:

$$V = r^d$$

This means, operations which are of the order of the volume will take exponential time to complete in the worst case. Even in operations that are not of the order of input space, the added dimensionality unfortunately poses a lot of problems. As an example, consider the Nearest Neighbor Query [2]. To find the nearest neighbors of a point P_1 in a **d** dimensional space, we evaluate the **similarity** of P_1 with every point P_2 in the space, and choose the few most similar points (nearest neighbors). Suppose we use the cosine similarity defined as:

$$\delta(x_1, x_2) = (x_1 \cdot x_2) \; / \; \left(\left| x_1 \right| \left| x_2 \right| \right)$$

[1]Email: nav.naveenvenkat@gmail.com | Web: naveenvenkat.com
[2]https://github.com/nmakes/curse-of-dimensionality

where x_1 and x_2 are the **d** dimensional vectors of the points P_1 and P_2, $(\cdot)$ denotes the scalar product and $||$ is L2 norm. Clearly, $O(d)$ time will be required for evaluating the cosine similarity. For **N** points, we obtain a worst case performance of $O(Nd)$. Essentially, the time required to evaluate the similarity is linear in the number of dimensions of the data space. Note the following observations:

1. As the dimensionality increases, sparsity of the data increases (or atleast, the possibility of sparse space increases), because the input space grows exponentially with respect to **d**.
2. It also degrades the performance of tasks which are dependent on the input space.
3. The similarity measure becomes meaningless as well, because all points will tend to be equi-similar with P_1, or in other words, the distinction between the similarities will reduce rapidly as **d** increases.

The dimensionality problem roots back to the gathering of the data, where one might obtain highly noisy (or redundant) dimensions which are barely informative. This only results in the increase in dimensions without any significant benefit because in most cases, a lesser number of dimensions might contain as much information as original dimensions. In such cases, proper data cleaning helps in obtaining better results. One solution to mitigate the curse of dimensionality is to transform the data from a higher dimensional space to a more useful lower dimensional space. Some traditional methods such as PCA [3] transform the data into the most informative space, thereby allowing the use of lesser dimensions which are almost as informative as the original data.

In this study, the properties of the dimensionality problem are analyzed and some methods to avoid or mitigate it are presented. In the following section we explore the various scenarios where the Curse of Dimensionality has visible effects, along with some experiments, and provide some measures to mitigate it.

II. Experiments & Observations

The experiments in this section are carried out on an Intel Core i7 7700HQ CPU.

A. Obtaining Cosine Similarity

As explained in section I, the effort required to measure cosine similarity will increase linearly and the similarity will blur out (will have less variation) as the number of dimensions increase. To evaluate the trends in the similarity calculation, we randomly generate **N = 1000** data points with increasing number of dimensions from **d = 10** to **1000** in steps of **10**. Each dimension of the data point is given a value between [0, 1]. We then choose a random point P_1 and measure its cosine similarity with the rest of the points. We perform this experiment for each dimension size (from 10 through 1000) and report the standard deviation of the similarity measure along with the running time for each experiment. Note that, since P_1 is selected at random, the results obtained can be generalized for any point.

The results are plotted in figure 1. We interpret the results as follows:

1. As expected, the running time shows (Fig. 1) a linear trend in the number of dimensions owing to the $O(Nd)$ time complexity ($N = 1000$).
2. The trend in the standard deviation shows that as dimensionality increases, the variation in the cosine similarity measure falls rapidly. This suggests that each point tends to be equi-similar with other points as the dimensionality increases, making it more and more difficult to distinguish between close points and closer points.

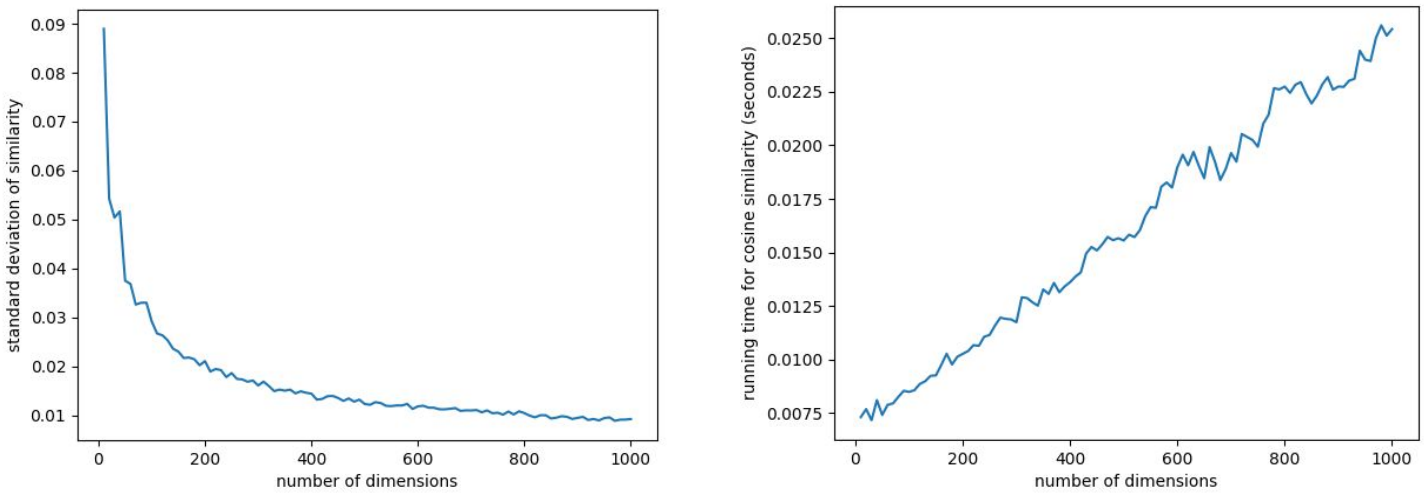

Fig 1: Trends in the standard deviation in cosine similarity (left) and running time to compute cosine similarity (right) with increasing dimensions.

Cosine similarity is a useful tool because not only does it provide an estimate of the relative orientation of two vectors in space, it is also computationally efficient, as the scalar product can be performed in parallel, allowing it to be used even for large inputs. However it fails to provide a good bias for nearest neighbor queries in high dimensions.

B. Evaluating L_p Norms

An order **p** norm is defined as:

$$||x||_p = \left(|x_1|^p + |x_2|^p + \ ... \ + |x_d|^p \right)^{1/p}$$

where $x = (x_1, x_2, \ ... \ x_d)$ is a vector with **d** dimensions. A prior condition of the norm is that it should follow the triangle inequality (thus for norms, $p \geq 1$). Traditionally, the euclidean norm (order 2) is used as a distance measure between two vectors (opposite to the concept of similarity). In general, the **distance** between two vectors x_1 and x_2 , can be defined by the L_p norm of the difference between the two vectors as follows:

$$\delta(x_1, x_2) = \left\| x_1 - x_2 \right\|_p$$

From the standard deviation trends in experiment A (Fig. 1), it is clear that the distinction between the points becomes blurred as the number of dimensions increase. This poor contrast leads to poor data visualization. A classic way to combat poor contrast is to apply a non-linear scaling on the values, which increases the separation between the values, thereby increasing the contrast. From experiment A, we know that higher the number of dimensions, the lesser will be the contrast in similarity (or conversely, distance). In a situation where the data has a large number of dimensions, and the reduction of dimensions is not possible, we wish to explore if a mitigation to the curse of dimensionality is possible. We compare the variability of the distance measure with respect to the order of the norm. We again use standard deviation to evaluate how well the norm is able to capture the contrast in distances.

We perform the experiment as in experiment A, by generating **N = 1000** points of dimensions **d = 10, 20, 30** and **40** each. For each dimension, we test the differentiating capacity of L_p norm from **p = 1** to **10** in steps of **0.1**. We report the standard deviation of the distances obtained in each test. The results of the tests are plotted in Figure 2.

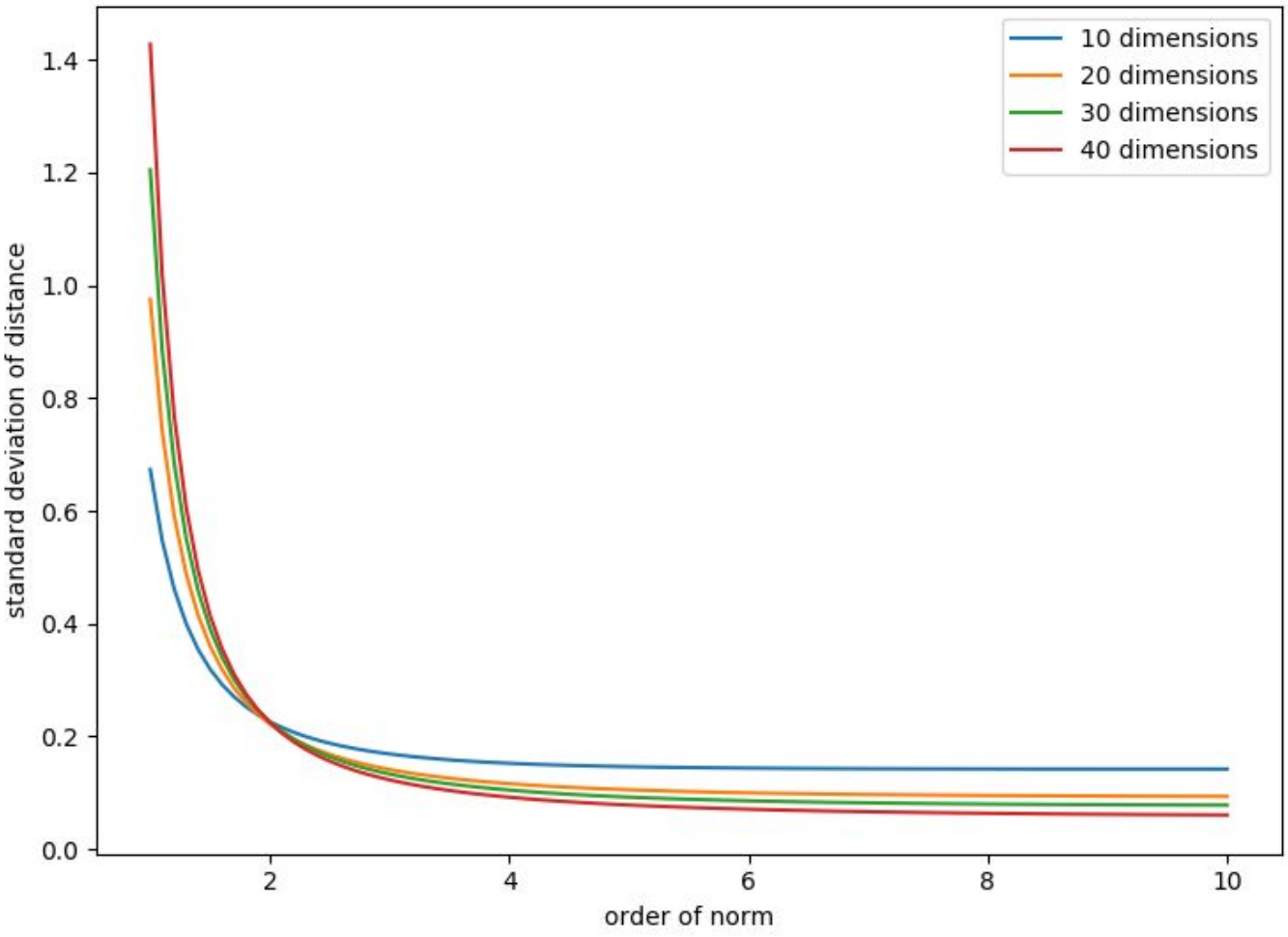

Fig 2: Trends in the standard deviation of the distance measure with increasing order of norm

Note the following observations:

1. Increasing the order of the norm reduces the standard deviation between the distances, which means it worsens the distinction capacity, as we argued in experiment A. Thus, a lower order norm yields a greater distinction between the values.
2. As seen with higher order norms, larger number of dimensions yield worse contrast than smaller number of dimensions (blue line indicates 10 dimensions, red line indicates 40 dimensions), suggesting that on increasing dimensions, the curse of dimensionality has a significant effect.

C. Irrelevant and Correlated Attributes

As the number of dimensions increase, the chance of having irrelevant and correlated attributes increases. **Irrelevant** dimensions are least informative [4] which essentially arise due to:

1. Very little variation in the attribute thereby providing no basis for distinction among the data points along that dimension.
2. High noise in the attribute, which although may add variance but no useful pattern will be latent because noise is inherently random.
3. Large number of missing data in the attribute, making the attribute useless for comparisons or decisions.

In any case, the irrelevant dimension will not play a major role in determining the property of a data point as it may not contain any knowledge about the data. Even if it does contain useful information, to retrieve it we need to either introduce appropriate transformation (1), or apply some noise filtering techniques (2), or gather more data (3).

On the other hand, **correlated** attributes are those that have a high degree of correlation. This implies, the existence of one attribute is unnecessary simply because another attribute captures similar patterns. This is a major problem with big data (for eg. large text corpora, long RNA sequences etc.) because here the curse of dimensionality dominates over the additional information captured in the dimension (if any). In such cases, domain knowledge is required which can highly reduce the dimensionality of the space. For instance, a classic problem in computational biology is to identify RNA-Protein interactions [5]. Proteins are long chains of **20** different amino acids. One representation is the **n**-gram frequency of the protein. In this representation, input space of the problem thus grows as 20^n. This is a huge space because sometimes the ideal values for **n** are between **5-10**. Here, it is unclear whether a dimension would be irrelevant or correlated to some other dimension. However, one solution explored by [5] exploits the domain knowledge to reduce dimensions, where chemically similar amino acids

are grouped together into **4** categories (hence reducing from 20 amino acids to 4 groups). Thus, the resulting space is reduced to 4^n, which is much smaller.

D. Diminishing Fraction of ε-Hypersphere and Optimization Performance

The nearest neighbor computation (explained in section I) is traditionally relaxed by searching in a local region instead of the complete space. For this reason, the space is divided into chunks and the nearest neighbor query is performed in the individual chunks. This method is faster, but faces some limitations. Suppose the **K** closest neighbors are to be found (for instance, in K-Nearest Neighbor classification algorithm), then they may not necessarily fall in the chunk (the chunk may have less than **K-1** data points). In such cases, the search needs to be carried out in multiple nearby chunks, and optionally the chunks can be merged after the search to speed up later queries. This is required because the possibility of a neighbor lying within a given euclidean distance (**r)** decreases with increasing sparsity of the data. Thus, several chunk searches may be necessary to find K closest neighbors. We can view this effect by taking the ratio of the volumes of a hypersphere and a hypercube. The volume of a hypercube is given in section I, while the volume of a hypersphere in **d** dimensions is:

$$V_d(r) = (\pi^{d/2} \cdot r^d) \,/\, \Gamma(d/2 + 1)$$

where Γ is Leonhard Euler's gamma function. We plot the ratio, of the volumes for a unit sphere ($r = 1$) and the enclosing hypercube ($side = 2r = 2$) for dimensions **d = 1** to **100**, in Figure 3.

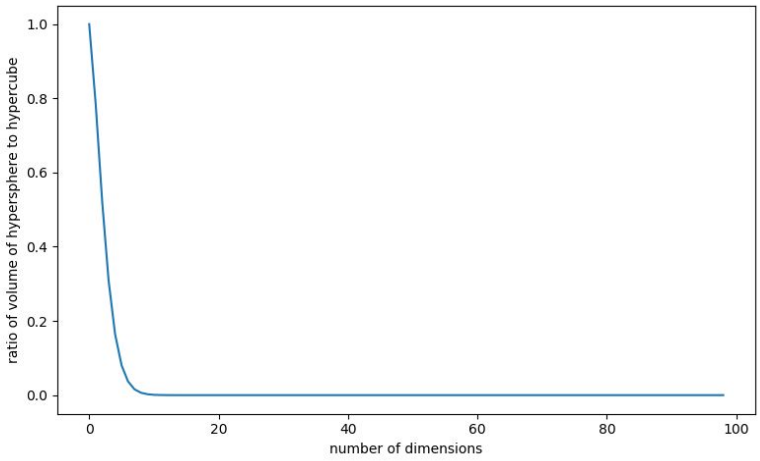

Fig 3: Ratio of the euclidean volume of a hypersphere to hypercube for dimensions from 1 to 100.

Clearly, the ratio falls rapidly and tends to 0 at higher dimensions. This suggests that most of the space is distributed around the corners of the hypercube, while the hypersphere takes very little

space. Figure 4 illustrates the empty space for 2 and 3 dimensions. Thus, the probability of finding a data point within a distance **r** (within the sphere) is much less (almost nil.) than the probability of finding it elsewhere for higher dimensions. This shows how the curse of dimensionality affects the space of the problem.

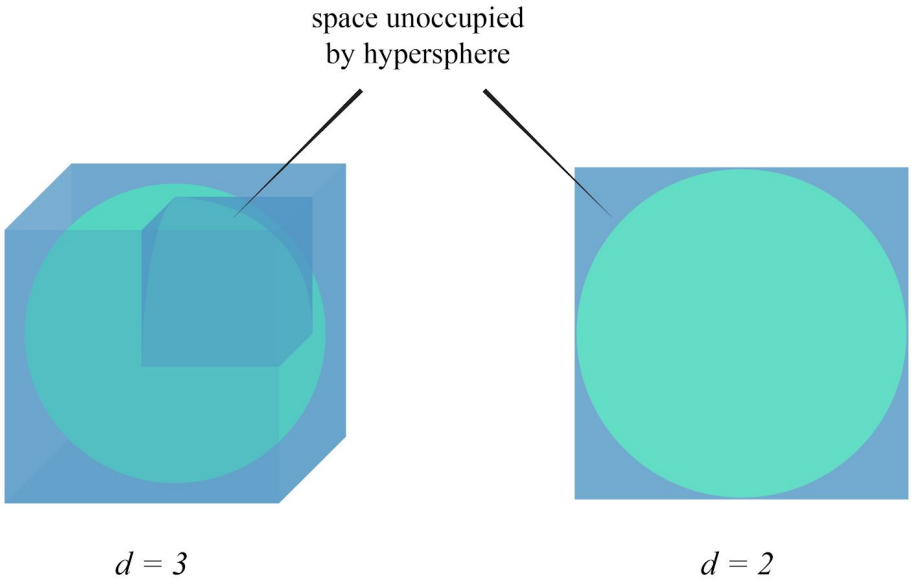

Fig 4: Illustration of the empty space for 2D and 3D spaces

This imposes a major hurdle in algorithms that depend upon the spatial locality of data points, such as clustering algorithms. A popular clustering algorithm, DBSCAN, is dependent upon the density of the points in space. More specifically, an epsilon volume around a candidate point is considered within which the density of points is inspected and the cluster is assigned accordingly. Due to the diminishing volume of the epsilon-hypersphere, we can expect the DBSCAN to poorly perform. Here, dimensionality reduction using mathematical tools like Principal Component Analysis helps obtain better convergence of such algorithms.

III. Conclusion

In this article, the nature of the Curse of Dimensionality (CoD) is studied. this effect is It is evident that higher dimensions come at their own cost. The exploding nature of spatial volume is at the forefront of the reasons for the curse of dimensionality. We observe that the effects of the CoD are easily pronounced with as little as a few tens of dimensions. This study provides a strong basis to argue against the ill effects caused by higher dimensional spaces. It is also observed that in some cases identifying the appropriate cause of poor performance is difficult, and prior domain knowledge may be needed. We suggest some improvements to mitigate the effects, with some of the prominent techniques transforming the data from the original space to a lower dimensional space.

References

[1] RE Bellman (1957). ***Dynamic programming***. Princeton University Press, Princeton, NJ, USA.
[2] NS Altman, (1992). ***An introduction to kernel and nearest-neighbor nonparametric regression***. The American Statistician.
[3] I Jolliffe (2011). ***Principal Component Analysis***. Lovric M. (eds) International Encyclopedia of Statistical Science. Springer, Berlin, Heidelberg.
[4] PN Tan, M Steinbach, V Kumar (2005). ***Introduction to Data Mining***. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.
[5] DS Jain, SR Gupte, R Aduri (2018). ***A Data Driven Model for Predicting RNA-Protein Interactions based on Gradient Boosting Machine***. Scientific Reports - Nature.