Curse of Dimensionality

Lecture Notes for Big Data Analytics

Faizad Ullah

February 2019

Contents

1 Problems with High Dimensional Data

High dimensional data refers to data with a large number of features, variables or dimensions often represented by the columns in a dataset, where each row is an instance or observation. Often, the number of features (columns) can exceed the number of instances (rows).

The term 'curse of dimensionality' coined by Richard Bellman refers to the difficulty of dynamic optimization with many variables. Broadly, the following issues are faced when working with high dimensional data:

1. Working with large dimensional data is computational challenging. Processing, Storing, Communication of high dimensional data require substantially more computational resources.

2. Large dimensional data is very hard to visualize and interpret

3. In addition, generally as number of features increases redundancy also increases - more noise is added to data than signal. This result in degradation of performance of all analytic methods. In these notes we focus on various manifestation of this issue.

The term is used to refer to different phenomena that arise in high dimensional data analytics. In particular we discuss the impact of high dimensionality of data on the two canonical proximity computation problems of distance matrix computation and nearest neighbor search. Recall we discussed that they are the building blocks of almost all data analytics.

We discuss the curse of high dimensionality in light of the distance matrix computation problem. The K-NN problem will be discussed in detail later.

We now discuss the issues that arise in the distance matrix computation with high dimensional data one by one.

2 Computational Complexity

Given a set X of m-dim vectors in $\mathbb{R}^m$, with $|X| = n$. A straight forward observation about running time of the brute force solution to compute the distance matrix D is $O(n^2 \times m)$, since for almost all distance measures computing $d(i, j)$ requires traversal of all coordinates of i and j. Note that this run-time grows linearly with dimensionality of data.

3 Data Sparsity

As the dimensionality of data increases, the relative input space covered by a fixed-size training set diminishes. Many methods require a sizeable number of examples/samples in every region of the space to support a hypothesis or train a generalizable model. Since the available data becomes very sparse because the volume of the space has increased so much, such methods that need statistical significance fail. For example, in machine learning a certain number of training data items is required to give meaning to a model (e.g. a classifier), which is not there in such a sparse space. Similarly, in statistical analysis a certain sample size is required to support a hypothesis, which is not available.

In class we discussed that suppose we have data for 1000 students performance (discretized scores of $0, 25, 50, 75, 100)\%$ in 2 courses c_1 and c_2. Then in total there are $5 \times 5 = 25$ different grade combinations. If the 1000 students are randomly distributed among each grade combination, then on average there are 40 students with each possible grade combination, which is a good enough sample to draw conclusions such as if, for a student, $grade(c_1) \leq 50$ and $grade(c_2) \geq 75$, then that student is likely to be a Math major. Now suppose there are 4 courses, then the number

of possible grades combination is $5^4 = 625$, and an average number of students per combination is 1.6. For 10 courses, this number reduces to 0.0001024. This means that almost all possible combinations are never observed.

4 Huge Search Space for Nearest Neighbor Search

A basic approach for nearest neighbor search is use a data structure that partition the input space into cells (grids or mesh). Having preprocessed the data (i.e. storing the dataset X in this grid structure). To find nearest neighbor(s) of a query point one locates the cell containing q. And return all points in X in that the cell and perhaps those in the 'neighboring' cells.

The search space for nearest neighbors in this data structure grow exponentially with the dimensionality of the data. i.e. The number of 'neighboring' cells to search for in 2-d is $3^2 = 9$, that in 3-d is 3^3, which in m-d the number of cells one need to examine is 3^m

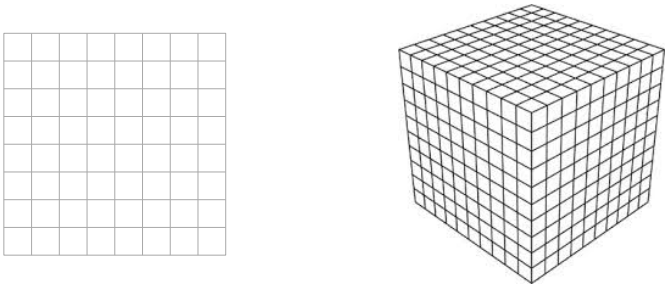

Figure 1 Partition the space into cells (grids or mesh) for large dimensions

Recall the approaches for nearest neighbor search. The grid (or lattice or mesh) could be non-uniform see for instance kd-tree, one of the most widely used data structures for nearest neighbor search, which works quite well when the dimension $n \leq 10$.

We demonstrate this phenomenon that higher dimensional neighborhood is very large and not local. i.e. the notion of nearest neighbor breaks down as follows.

Suppose n points in X are chosen uniformly at random from $[0,1]^m$ (m-cube). For the query point q grow a hypercube around q to contain f fraction of points ($k = fn$) in X. We show that this cube (the search space for q) grows very large (covering almost the whole input space) in large dimension. The Expected length of the edge of the search cube $E_m(f) = (f^{1/m}$. i.e. in $10d$ to get 10% points around q need cube with edge length 0.8 (which is 80% of the whole cube, the input space). Similarly, to get only 1% points one needs to extend the search cube by 0.63 units along each dimension

We give a concrete example to demonstrate this phenomenon of non-locality of higher dimensional neighborhoods

Suppose 5000 points are randomly placed in $[0,1]^m$. Let $q = \mathbf{0}$

- In 1d we must go a distance of $5/5000 = 0.001$ on average to capture 5 NN (i.e to to capture 5 of those random input points for a typical q, we need to grow a search cube to capture only .1% of the input cube)

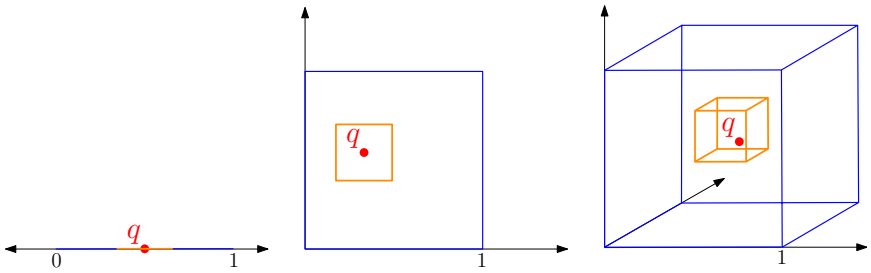

Figure 2 1-d (left), 2-d (middle) and 3-d (right) search spaces example

- In 2d, on average we must go a distance $\sqrt{5/5000} = 0.031$ units along both dimensions to get 5 nearest neighbors points (about 3% of the whole cube)
- In 3d, on average we must go $\sqrt[3]{0.001} = 0.1 = 10\%$ of the total (unit) length in each of the 3 dimensions
- In 4d, we must go $\sqrt[4]{0.001} = 0.177 = 17.7\%$ of unit length
- In 10d, we must go 50.1% of unit length along each dimension
- In md, we must go $(5/5000)^{1/m}$ along each dimension

To express this phenomenon the phrase "in high dimensional space nobody can hear you scream" is used.

5 Diminishing Volume of m-ball

A manifestation of this phenomenon that points in higher dimensions are isolated is the diminishing relative volume of the m-ball in m-cube

There is another way one can look at this issue with high dimensional space that is a big hurdle for the fixed radius nearest neighbors search problem. The problem is that in very large dimensions, this ball (the output of the problem) is essentially empty. This is sometimes referred to as in high dimensions, every point is an outlier. This is also referred to as high dimensional space is lonely.

The m-ball (m-d hypersphere) of radius r centered at origin is defined as

$$B_{m,r} := \left\{\mathbf{x} \in \mathbb{R}^m : d(\mathbf{x}, \mathbf{0} \leq r) \implies \|\mathbf{x}\|_2 \leq r\right\}.$$

The m-dimensional volume of m-ball of radius r in m-dimensional Euclidean space is:

$$\text{Volume of } B_{m,r}: \qquad V_m(r) = \frac{\pi^{m/2}}{\Gamma(m/2+1)}\, r^m$$

$\Gamma(\cdot)$ essentially is factorial of fractional numbers

For our purposes

$$V_m(r) = \frac{\pi^{m/2}}{m/2!} r^m \qquad \text{For simplicity assume } m \text{ is even}$$

The m-cube (m-d hypercube) is the set $[-1, 1]^m$ (note edge length is 2)

$$\text{Volume of } m\text{-cube:} \qquad 2^m$$

In m-d ratio of volume of unit m-Ball to that of m-cube (edge length 2) is

$$\frac{\pi^{m/2}/m/2!}{2^m} \text{ approaches 0 very fast}$$

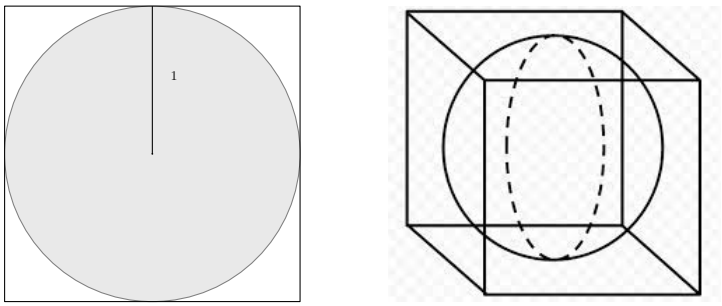

Figure 3 In 2-d (left) ratio of volume of unit m-ball is higher than that of 3-d (right) to that of unit m-cube, and approaches to 0 very fast in higher dimensions

Observe the ratio in the following table.

dim m	volume of m-ball	volume of m-cube	ratio
2	π	2^2	~ 0.785
3	$4/3\pi$	2^3	~ 0.523
4	$\pi^2/2$	2^4	~ 0.308
6	$\pi^3/6$	2^6	~ 0.080
m	$\frac{\pi^{m/2}}{m/2!}$	2^m	$\to 0$

Ratio of volumes of unit m-Ball and $[-1, 1]^m$

$$\frac{\pi^{m/2}/m/2!}{2^m}$$

Let say we select a point x at random in the $[-1, 1]^n$ (cube centered at the origin) cube, what is the probability that x is the unit n ball (inscribed). This probability is exactly equal to the ratio of the volume of the unit ball to the volume of the cube.

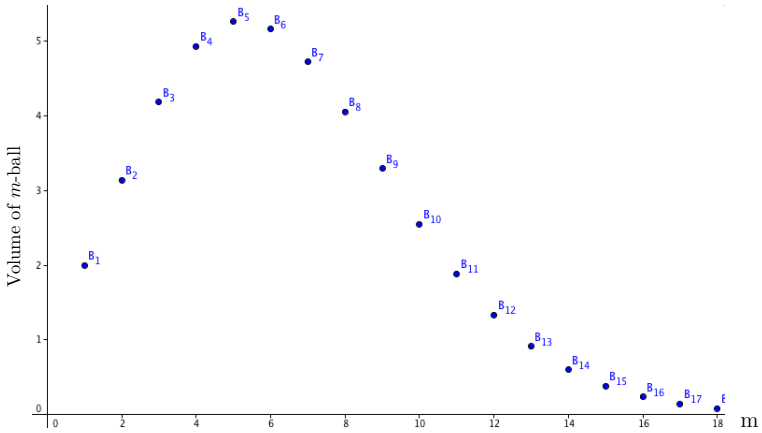

Figure 4 Volume of m-ball (denoted by B_m) increases from $m = 1$ to $m = 5$, afterwards it sharply declines and get close to 0 for $m > 20$.

Hence the volume of the unit sphere compared to the cube is diminishing. This means that in very high dimensions, if we are doing a nearest neighbor search, (say the fixed radius version), it is empty, almost no point is within distance r. In other words for a fixed query point q, if we choose n points at random in $\mathbb{R}^m$ almost no point will be within r distance to q, equivalently almost all will be at distance more than r. So nearest neighbor (or distance for that matter) lose all its effectiveness.

- In higher dimensions all the volume is in 'corners'
- Points in high dimensional spaces are isolated (empty surrounding)
- The probability that a randomly generated point is within r radius of q approaches 0 as dimensionality increases
- The probability of a close nearest neighbor in a data set is very small

There is a caveat that we must keep in mind for this and the following issues that the real datasets are not random.

However if a dataset exhibit this phenomenon that the issue has be overcome by getting a larger training set (exponential in m). One way to look at this is as follows.

To cover $[-1, 1]^m$ with $B_{m,1}$'s, the number of balls n must be

$$n \geq \frac{2^m}{V_m(1)} = \frac{2^m}{\pi^{m/2}/_{m/2!}} = \frac{^{m}/_{2}!\, 2^m}{\pi^{m/2}} \overset{m\to\infty}{\sim} \sqrt{m\pi}\left(\frac{m2^{m/2}}{2\pi e}\right)^{m/2}$$

For $m = 16$ (a very small number) this n is substantially larger than 2^{58}

6 Instability of Nearest Neighbor

A qualitative problem in higher dimensional space resulting from the phenomenon of empty neighborhood is that the notion of nearest neighbor breaks down. This means that there is substantial difference or contrast between nearest and farthest neighbors of a point q.

A **nearest neighbor query is ϵ-unstable** ($\epsilon > 0$), if the distance from q and most other points are at most $(1 + \epsilon)$ times the distance from q to its 1NN.

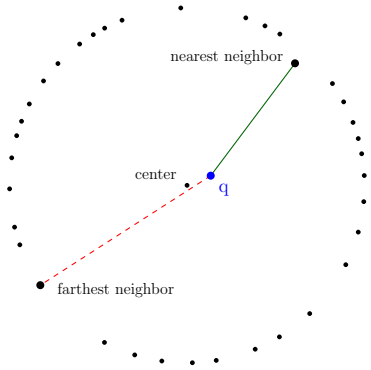

Figure 5 No difference (contrast) between nearest and farthest neighbors in higher dimensions

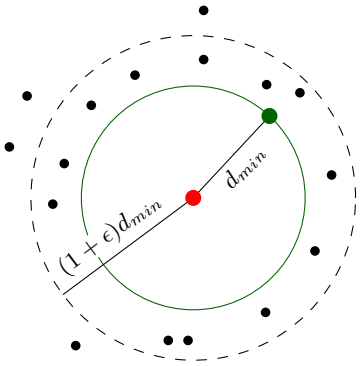

Figure 6 Higher dimension causes ϵ-unstable ($\epsilon > 0$) nearest neighbor query

We show that as dimensionality increases the probability of all nearest neighbors queries becoming unstable increase (by showing the distance concentration phenomenon)

7 Distance Concentration

Another facet of the curse of dimensionality is the phenomenon of distance concentration.

Assume points in $\mathbb{R}^m$ and ℓ_2 distance measure. As m increases, almost all pairs of points have their ℓ_2 distances (i) similar to distance of other pairs and (ii) very high. Consequently, the distance measure loses its meaning, and since proximity measures is the building block of data analytics, as we discussed earlier, when it becomes meaningless the building collapses. For example, in the K-NN problem, if all distances are similar and high, the nearest neighbor is as good as the farthest neighbor and it becomes difficult to build clusters since there is no justification to group a pair of points and not another.

It can be said that the normalized distance is close to 1, so both factors that distances are high and similar are encompassed. We demonstrate it by observing distribution of pairwise distances for n points in $\mathbb{R}^m$.

7.1 Analytical Bounds

Suppose we generate a set $\mathcal{X}$ of random vectors in m-dimensional unit cube with $|\mathcal{X}| = n$, i.e. n points in $[0,1]^m$. Let $x = (x_1, \ldots, x_m)$ and $y = (y_1, \ldots, y_m)$ be two such vectors. The Euclidean distance ℓ_2 between x and y is given by

$$d(x,y) = \|x - y\|_2 = \sqrt{\sum_{i=1}^{m} (x_i - y_i)^2}.$$

The maximum possible distance b/w a pair $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ is $d(\mathbf{x}, \mathbf{y}) \leq \sqrt{m}$. For simplicity we consider the squared distance (to get rid of square root) i.e.

$$d^2(x,y) = \|x - y\|_2^2 = \sum_{i=1}^{m} (x_i - y_i)^2.$$

Thus,

$$d^2(\mathbf{x}, \mathbf{y}) := \|\mathbf{x} - \mathbf{y}\|^2 \leq m$$

For a fixed coordinate $i < m$, We have that

$$Pr(|x_i - y_i| \geq 1/4) > 1/2. \tag{1}$$

To see this, note that $Pr(|x_i - y_i| \geq 1/4) = Pr(\{x_i - y_i \geq 1/4\} \cup \{v_i - u_i \geq 1/4\})$. Since the joint distribution of x_i and y_i is uniform on $[0,1]^2$, this event correspond to area of the lower right triangle and upper left triangle (below the line $y_i = x_i - 1/4$ and above the line $y_i = x_i + 1/4$, the shaded regions in Figure 7). These two triangles stacked together, is a square of length $3/4$, and its area (and hence $Pr(|u_i - v_i| \geq 1/4)$) is $9/16$ which is greater then $1/2$.

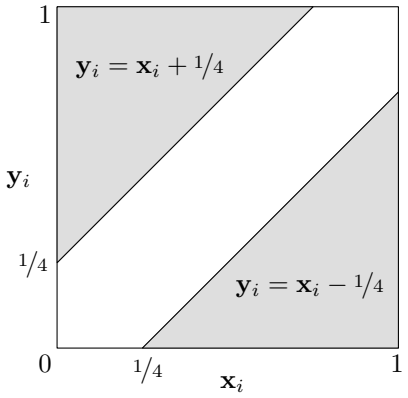

Figure 7

Using this we will argue that when dimensions are large then almost all distances are large.

The following is a very useful and elegant trick, we will use it perhaps many timed. It is imperative that you master it, so think about it and keep thinking until it is clear (without worrying about technicalities like exact calculations etc.).

For fixed $x, y \in \mathbb{R}^m$ chosen as above, let V_i be the indicator random variable for the event that $|x_i - y_i| \geq 1/4$. i.e. if coordinate difference is big.

$$V_i = \begin{cases} 1 & \text{if } |x_i - y_i| \geq 1/4 \\ 0 & \text{otherwise} \end{cases}$$

We know that $E[V_i] = Pr(V_i = 1) > 1/2$ by Equation (1). Let $V = \sum_{i=1}^{m} V_i = \left|\left\{i : |\mathbf{x}_i - \mathbf{y}_i| \geq 1/4\right\}\right|$, i.e. V is the counter to see how many coordinates have big difference. By linearity of

expectation, we know that $E(V) = m/2$, which means that on average $m/2$ summands of the d^2 sum will be at least $1/16$.

In order to show concentration around this mean, which implies that with very high probability the distance is large for this pair, we will use Chernoff bound.

Theorem 1 (Chernoff Bound (tail inequality)). *Let $X_1, X_2, \ldots, X_n$ be independent Bernoulli random variables. Let $S = X_1 + X_2 + \ldots + X_n$, and let $E(S) = \mu$. The loose Chernoff bounds are stated as follows:*

- $Pr(S \geq (1+\delta)\mu) \leq e^{\frac{-\delta^2\mu}{3}}, \quad for\ 0 < \delta < 1$
- $Pr(S \geq (1+\delta)\mu) \leq e^{\frac{-\delta\mu}{3}}, \quad for\ \delta > 1$
- $Pr(S \leq (1-\delta)\mu) \leq e^{\frac{-\delta^2\mu}{2}}, \quad for\ 0 < \delta < 1$

Using the Chernoff Bound with $\delta = 1/2$:

$$Pr(V < m/4) = Pr(V < (1 - 1/2)m/2) < e^{-m/16}$$

So almost surely, at least $m/4$ coordinate differences are at least $1/4$, and hence squared distance is at least $m/64$. So, the probability that a given pair is far, i.e. its squared distance is more than $m/64$ is at least $(1 - e^{-m/16})$.

$$Pr\big[V \geq \frac{m}{4}\big] \geq 1 - e^{-\frac{m}{16}}$$

In other words, for fixed $\mathbf{x}, \mathbf{y}$: $Pr\big[\|\mathbf{x} - \mathbf{y}\|^2 \geq \frac{m}{64}\big] \geq 1 - e^{-\frac{m}{16}}$

Now we want to find what is the probability that all $\binom{n}{2}$ pairs are far. This probability is equal to 1 minus the probability that some pair is close (squared distance $m/64$). This latter probability is less than the sum of probabilities of individual pair closeness (union bound).

$$Pr\left(\text{all } \binom{n}{2} \text{ pairs are far}\right) \geq 1 - \sum_{\text{pairs}} e^{-m/16} = 1 - \binom{n}{2} e^{-m/16}$$

Solving the above inequality such that the probability is at least $1/2$, then as long as $m = \Omega(\log n)$, then with high probability for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, we have $d^2(\mathbf{x}, \mathbf{y}) \geq \frac{m}{64}$. This means all pairs are far, i.e. $d(\mathbf{x}, \mathbf{y}) \geq \sqrt{m}/8$.

Here is the result of simulation of this distance concentration phenomenon. In your homework you were supposed to observe this for varying parameters.

8 Angle Concentration

In large dimensions (at least for random points) the distance measure (at least ℓ_2 distance) is more or less meaningless. Then, can we use cosine distance? We will show that the same concentration phenomenon is observed for pairwise angles.

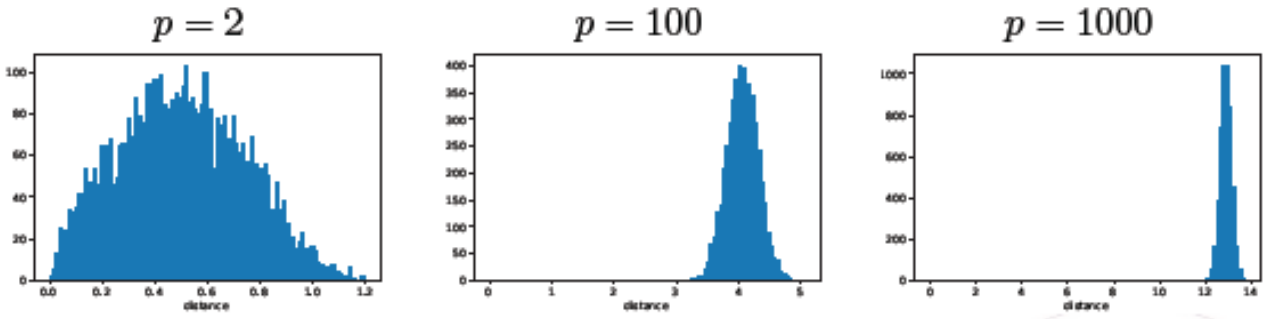

Figure: Histograms of pairwise-distances between $n = 100$ points sampled uniformly in the hypercube $[0, 1]^p$

A pair of vectors $\mathbf{x}, \mathbf{y}$ is orthogonal if $\mathbf{x} \cdot \mathbf{y} = 0$, $\theta_{x,y} = 90°$. The maximum number of such pairwise orthogonal vectors in $\mathbb{R}^2$ is 2 and in $\mathbb{R}^3$ is 3. In $\mathbb{R}^m$, the maximum number of pairwise almost orthogonal vectors ($\mathbf{x} \cdot \mathbf{y} \leq \epsilon$, $\theta_{x,y} = 90° \pm \epsilon$) is $e^{\Omega(m)}$.

8.1 Generating Random Direction in $\mathbb{R}^m$

Choosing a random direction (equivalently, a random unit vector) is not a straight forward task. We think of this as choosing random unit vectors (normalized) in $\mathbb{R}^m$. W will also need this later for dimensionality reduction and also for Random Hyperplanes based LSH for Cosine distance and Random Projections based LSH for Euclidean distance.

An immediate way of picking a random vector in $\mathbb{R}^m$ is to choose a random point in $\mathbf{v} \in [-1, 1]^m$ and normalize it as $\hat{\mathbf{v}} = \mathbf{v}/\|\mathbf{v}\|$. For example, in $2D$ ($\mathbb{R}^2$), generate a point in the unit square and then normalize it. However, this doesn't produce all directions uniformly at random, i.e. let $v = (v_1, v_2)$ where v_1 and v_2 are randomly chosen from $[0, 1]$. This picks a random vector in the unit square (or cube as we saw earlier), we can normalize v as $\hat{v} = \frac{v}{||v||_2} = \frac{v}{\sqrt{v_1^2+v_2^2}}$.

As you can see in Figure 8, the distribution of randomly generated direction is skewed towards the diagonal directions.

A quick fix to this method is the algorithm of George Marsaglia and Arif Zaman, i.e. to generate a random vector in the unit cube, but if the vector is outside the unit ball, then discard it, (i.e. if $v_1^2+v_2^2+\ldots v_m^2 > 1$, then discard it), so we only consider points within the unit ball. Then, when normalized, each point on the surface of the unit ball will be equally likely. However, this can be computationally expensive. Recall the problem of the diminishing volume of the m-ball, which implies that all points points lie outside the unit m-ball and thus will

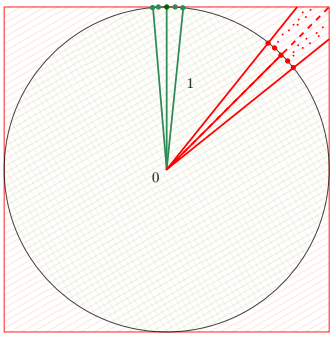

Figure 8 The distribution is skewed towards the diagonal directions

have to be discarded.

In $2D$, an easier way is to get a random direction is to generate a random number between $[0, 2\pi]$ (randomly choose an angle) and use standard trigonometric calculations to create a unit vector. The former method is computationally more expensive while the latter only works for $2D$.

The correct way to generate random directions (or random unit vectors equivalent to generating random points on the surface of the unit m-ball) is to use the spherical symmetry of the standard normal distribution. We generate m dimensional vector $v = (v_1, \ldots, v_m)$ by picking each of v_i independently from the standard normal distribution $\mathcal{N}(0, 1)$ and then normalize v by $\|v\|_2$ as earlier. Such a random vector in $\mathbb{R}^m$ is known to be uniformly distributed over the surface of unit ball B_m, as can be seen in Figure 9.

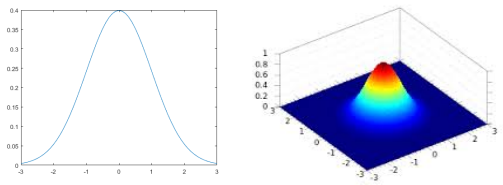

Figure 9 Generating random points on the surface of the unit m-ball

An approximate method to generate a random direction in $\mathbb{R}^m$ is to select the vector $v = (v_1, \ldots, v_m)$ by choosing $v_i \in \{-1, 1\}$ independently and equally likely. v is then normalized by $\|v\|_2 = \sqrt{m}$. This generate directions towards corners of the m-cubes $[-1, 1]^m$ as shown in Figure 10.

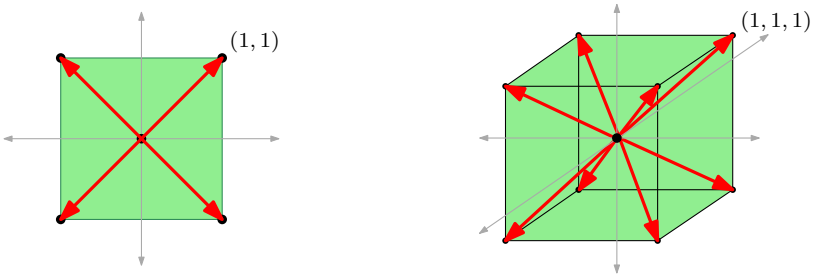

Figure 10 Generate directions towards corners of the m-cubes

The approximation is quite good in large dimensions and we will use it extensively in dimensionality reduction. In fact, we will discuss both of these techniques during the dimensionality reduction module. In order to see that almost all vectors are almost orthogonal in high dimensions, we will use analytical bounds to show that in expectation, every pair of random high dimensional vectors is orthogonal [1].

8.2 Analytical Bounds

Similar to distance concentration analytical bounds, Suppose we generate a set $\mathcal{X}$ of random vectors in m-dimensional unit cube with $|\mathcal{X}| = n$, i.e. n points in $[0, 1]^m$. Suppose these are random points on the surface of B_m, i.e. they are just directions chosen uniformly at randomly from $\{-1, 1\}^m$ and normalized as discussed above.

Recall that unit vectors x and y are almost orthogonal mean that $\theta_{x,y}$ the angle between them is about $90°$, or $\cos\theta_{x,y} = \langle x, y\rangle = x \cdot y = \sum_{i=1}^{m} \sim 0$ (as the denominator is 1).

For a fixed $\mathbf{x}$ and a random unit vector (chosen as in previous section) $\mathbf{y}$ in $\mathbb{R}^m$, let $V_i = \mathbf{x}_i\mathbf{y}_i$ and let $V = \sum_{i=1}^{m} V_i = \cos\theta_{\mathbf{x},\mathbf{y}}$

Lemma 2. *$E[V_i] = 0$ and $\frac{-\mathbf{x}_i}{m} \leq V_i \leq \frac{\mathbf{x}_i}{m}$.*

This implies that on average, $\mathbf{x}$ is orthogonal to any $\mathbf{y}$. To bound the probability of the deviation of V from its mean, we use the Hoefding's inequality.

Theorem 3 (Hoeffding's inequality)**.** *Let X_i be random variables bounded by the interval $[a_i, b_i]$ and let $S = \sum_{i=1}^{n} X_i$. Then*

$$Pr(|S - E[S] \geq t) \leq 2\exp\left(-\frac{2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right)$$

For $S = V$:

$$Pr(V \geq \epsilon) \leq 2\exp\left(-\frac{2\epsilon^2}{\sum_{i=1}^{m}(2x_i/m)^2}\right) = 2e^{-\epsilon^2 m/2}$$

Thus, with very small probability, a pair of vectors deviates from orthogonality, i.e. the cosine of the angle between them is more than ϵ.

We can use union bound to compute the probability that no pair deviates from orthogonality which is equal to 1 minus the probability that some pair $\mathbf{x}, \mathbf{y}$ is not orthogonal, i.e. $\cos(\theta_{x,y}) \geq \epsilon$.

$$Pr(\forall\, x, y \in \mathcal{X} : \cos(\theta_{x,y}) < \epsilon) \geq 1 - \sum_{\text{pairs}} 2e^{-\epsilon^2 m/2} = 1 - \binom{n}{2} 2e^{-\epsilon^2 m/2}.$$

For $m = \frac{6}{\epsilon^2}(\ln n)$, we get that

$$Pr(\forall\, x, y \in \mathcal{X} : \cos(\theta_{x,y}) < \epsilon) \geq 1 - n^2 e^{-\frac{\epsilon^2 m}{2}} = 1 - \frac{1}{n}.$$

Thus, almost surely all vectors are mutually orthogonal (for $0 < \epsilon < 1$).

References

[1] Achlioptas D. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 66(4):671–687, 2003.