CMSE 890-002: Mathematics of Deep Learning, MSU, Spring 2020

Lecture 08: Curse of Dimensionality and k-NN

January 29, 2020

Lecturer: Matthew Hirn

4.3 Curse of dimensionality

We have now examined two models closely: linear models and k-nearest neighbors. In the previous section, Section 4.2, we observed that linear models complexity scales with the dimension d of our data, but the variance only scales linearly in d. Thus linear models are stable. On the other hand, if the mapping $x \mapsto y$ is not linear, a machine learned linear model will not be able to capture the relationship between data point x and label y, and there will be a potentially large bias in the model.

For k-nearest neighbors the situation is a bit more subtle. Figures 11 and 12 would seem to indicate that it is superior to the linear model and in general a good choice. Indeed, without knowing the underlying data generation process, which was highly nonlinear, k-nearest neighbors resulted in a model that was nearly as good as the naive Bayes model, which is the best one can do under the circumstances of our framework. The main issue from the analysis of Section 4.2 is that in order to reduce the variance of the k-nearest neighbor model we must select a reasonably large k (certainly not $k = 1$, see Figure 10). However, increasing k may increase the bias, as we select more and more points that are further away from the point who's label we are trying to predict. It would seem, though, that if we have a large amount of data, this issue is removed as we can select a large k without having to incorporate points that are far away from the central point x into its neighborhood $N_k(x)$. This intuition works in low dimensions but breaks down in high dimensions. While there are many manifestations of this problem, they are all generally referred to as the *curse of dimensionality*. In what follows we give a few examples.

Example 4.5. Here is one manifestation. Consider a d-dimensional unit cube, $\mathcal{X} = Q \subset \mathbb{R}^d$,

$$Q = [0,1]^d = \underbrace{[0,1] \times \cdots \times [0,1]}_{d \text{ times}}$$

Suppose our test point x is the corner, $x = (0, \ldots, 0)$, and that our training set is distributed uniformly in Q, meaning that $p_X(x) = 1$ for all $x \in Q$. A related method to k-nearest neighbors is to simply take all points within a geometric neighborhood of x and average their labels to obtain an estimate for the label of x. For example, we could take a sub-cube $Q_x \subseteq Q$, and average all the labels y_i of training points $x_i \in Q_x$ to obtain an estimate for the label of x:

$$f(x;\theta) = \text{Avg}\{y_i : x_i \in Q_x\}\,. \tag{14}$$

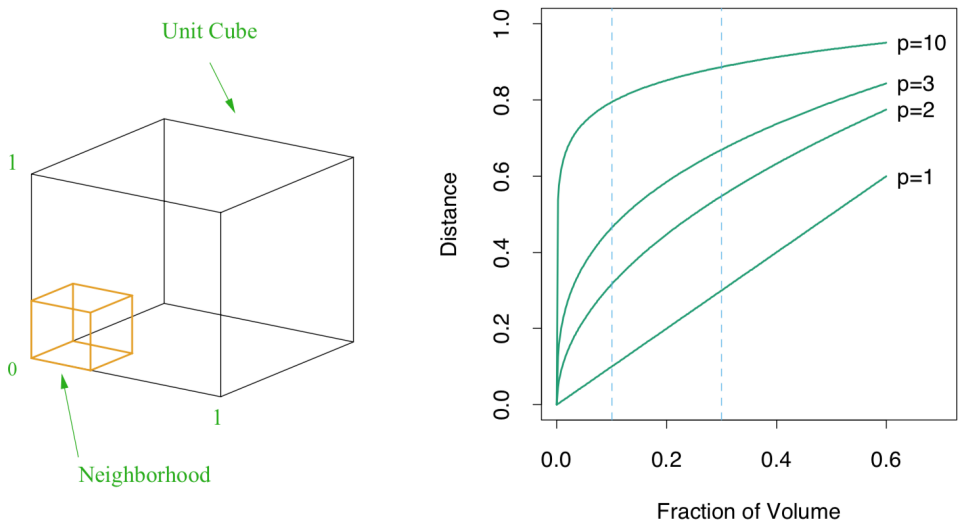


Figure 16: *One illustration of the curse of dimensionality. Left: A neighborhood cube Q_x (orange) as a subset of the unit cube Q. Right: The horizontal axis is the fraction of volume r that the neighborhood cube contains. The vertical axis is the required side length $e_d(r)$ of the neighborhood cube Q_x. Curves plotting $e_d(r)$ for $d = 1, 2, 3, 10$ are shown (note: in the figure, $p = d$).*

The parameters θ of this model have to do with how we construct the cube Q_x. Let us suppose we want to capture a fraction r of the volume of Q, which is one (i.e., $\text{vol}(Q) = 1$). How long must the edge of Q_x be? Let's call this edge length $e_d(r)$, since it depends on the fraction of the volume r we want to capture and the dimension d of the data space. In one dimension, it is clear that $e_1(r) = r$. However, in d-dimensions, we have

$$e_d(r) = r^{1/d}\,.$$

This is decidedly less favorable. Indeed, as an example, in 10 dimensions we have:

$$e_{10}(0.01) = 0.63 \quad \text{and} \quad e_{10}(0.1) = 0.80\,,$$

meaning that to capture just 1% of the volume in 10 dimensions we need an edge length of 0.63, and to capture 10% of the volume in 10 dimensions, we need an edge length of 0.80. Remember the side length of the cube Q is just 1.0! Therefore, what we thought was a local neighborhood due to our intuition about how things work in low dimensions, turns out, in fact, to be a very large neighborhood in high dimensions. Figure 16 illustrates this principle.

Example 4.6. Suppose instead we consider spherical neighborhoods, as is often the case. We consider a model similar to (14) but instead average the labels within a sphere of radius r centered at x:

$$f(x; r) = \text{Avg}\{y_i : \|x - x_i\|_2 \leq r\}\,. \tag{15}$$

Let us again suppose that our data space is $\mathcal{X} = Q = [0, 1]^d$, the unit cube. Suppose further that $x = (1/2, \ldots, 1/2)$, the center point of the cube, and we take the largest radius r

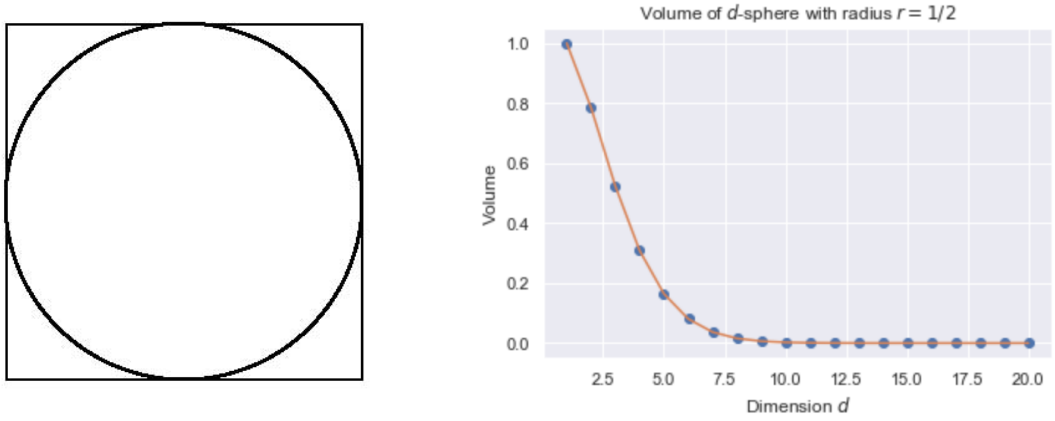
possible, which is $r = 1/2$. If the data points are sampled uniformly from Q, then the odds of no training points being within $1/2$ of x are

$$\mathbb{P}(x_i \notin B\,,\, \forall\, 1 \leq i \leq N) = \left(1 - \frac{\mathrm{vol}(B)}{\mathrm{vol}(Q)}\right)^N = (1 - \mathrm{vol}(B))^N\,,$$

where B is the ball of radius $1/2$ centered at $x = (1/2, \ldots, 1/2)$ and we used the fact that $\mathrm{vol}(Q) = 1$. The volume of a ball of radius r, B_r^d, in d-dimensions, is

$$\mathrm{vol}(B_r^d) = \frac{\pi^{d/2}}{\Gamma(d/2+1)} r^d\,.$$

In two dimensions, $\mathrm{vol}(B) \approx 0.79$, and so with $N = 100$ training points sampled from Q, the odds that no training points are in B is approximately $(0.21)^{100} \approx 0$; in other words, we are nearly guaranteed to have some training points close to x. Indeed, Figure 17a plots a circle inscribed in a square in two dimensions, and we see that the area of the circle dwarfs the area of the corners inside the square, but outside the circle.



(a) *A circle inscribed in a square.*

(b) *Volume of the d-dimensional ball of radius* $1/2$ *as a function of the dimension* d.

Figure 17: *Another illustration of the curse of dimensionality. Left: In low dimensions, it is likely that a test point x centered in a box will have training points sampled within a small radius of the point x. Right: In high dimensions, however, the volume of the d-sphere of radius $r = 1/2$ inscribed in the unit cube is dwarfed by the cube's unit volume, indicating that almost certainly the training points will be be situated in the "corners" of the cube and thus far from the test point x centered in the middle of the cube.*

On the other hand, in high dimensions the volume of the ball B rapidly decreases while the volume of the cube Q remains fixed at one; see Figure 17b. It thus follows that it is very unlikely for a training point to lie within the ball of radius $1/2$ centered at the test point x, making the model (15) useless. Indeed, if $d = 10$ and $N = 100$, then the odds of no training points lying within B are over 90%. When $d = 20$, it is essentially guaranteed!

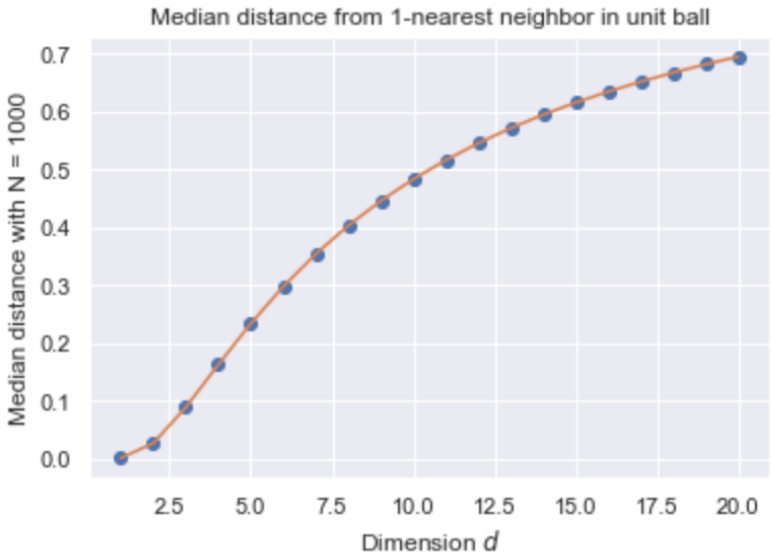


Figure 18: *The median distance of the 1-nearest neighbor to the origin in the unit ball as a function of dimension d, assuming the training points are sampled from the uniform distribution. As the dimension d increases, even the 1-nearest neighbor becomes very far from the test point at the origin.*

Example 4.7. Instead of sampling points from the cube Q, let us restrict to the ball B. Surely in this case the situation must be more favorable? In fact the answer is still no. Let us assume now that B has a radius of $r = 1$ and is centered at the origin. We consider the test point $x = (0, \ldots, 0)$ at the origin, and sample training points $\{x_1, \ldots, x_N\}$ from $\mathcal{X} = B$ according to the uniform distribution over B (so again p_X is constant). We are interested in the 1-nearest neighbor distance from x. One can show that over all possible draws of the training set, the median distance from the origin x to closest training point is:

$$\text{dist}_{\text{1-NN}}(d, N) = \left(1 - (1/2)^{1/N}\right)^{1/d} .$$

As with the other examples, in low dimensions we are fine. Indeed, for $d = 2$ and $N = 1000$, we have $\text{dist}_{\text{1-NN}}(2, 1000) \approx 0.03$. On the other hand, if we go to $d = 20$ dimensions, the situation becomes far worse, as $\text{dist}_{\text{1-NN}}(20, 1000) \approx 0.70$! Keep in mind, the distance to the boundary is one. Figure 18 plots $\text{dist}_{\text{1-NN}}(d, 1000)$ for $1 \leq d \leq 20$. Since this is the 1-nearest neighbor distance, it means that all k-nearest neighbors will be far from x in high dimensions, thus leading to poor models.

Example 4.8. In this example we can see the ramifications of the analysis contained in the previous examples. Suppose that $\mathcal{X} = [-1, 1]^d$, the cube of side-length two centered at the origin. Suppose additionally that labels y are derived from x according to:

$$y = F(x) = e^{-8\|x\|_2^2} ,$$

with no measurement error. We are going to use a 1-nearest neighbor model to estimate the label y of new test points. Let us consider a test point $x = (0, \ldots, 0)$ at the origin, which has label $f((0, \ldots, 0)) = 1$. Draw a training set $T = \{(x_i, y_i)\}_{i=1}^N$ and let $x_0 \in T$ be the point closest to x, and let $y_0 = F(x_0) = e^{-8\|x_0\|_2^2}$, which is the estimate for the label of x. Then using the bias-variance decomposition (Theorem 4.2), the expected test error at x is:

$$\text{Err}(\text{1-NN}, x) = \underbrace{(1 - \mathbb{E}_T[y_0])^2}_{\text{bias}} + \underbrace{\mathbb{E}_T\left[(y_0 - \mathbb{E}_T[y_0])^2\right]}_{\text{variance}}.$$

Suppose that the number of training points is $N = 1000$. In this case, the model will be biased because we know that $y_0 \leq 1$ no matter what. And indeed, the bias error will dominate, and grow large as the dimension d increases, since like in Example 4.7 and Figure 18 the median (and mean) distance of the 1-nearest neighbor from the origin will grow with the dimension. By dimension $d = 10$, more than 99% of the training samples will be, on average, at a distance greater than 0.5 from the origin, leading to a severe underestimate of the label since the $F(x) = e^{-8\|x\|_2^2}$ decays very rapidly; see Figure 19. Note that the bias does not always dominate. For example, if $F(x)$ only depends on a few dimensions of the data, e.g., $F(x) = (1/2) \cdot (x(1) + 1)^3$, then the variance error will dominate and will grow rapidly with the dimension; see Figure 20.

Another way of thinking about the curse of dimensionality is to consider how many training points N would be required to avoid it. Indeed, suppose in one dimension we require $N = 100$ training points to densely sample the space $\mathcal{X}$, e.g., $\mathcal{X} = [0, 1]$. Then to have the same sampling density for $\mathcal{X} = Q = [0, 1]^d$, we would require 100^d training points!

Now, in practice, we are very rarely confronted with a supervised learning problem in a high dimensional cube Q or ball B with a uniform sampling density. Indeed, consider the MNIST data base, consisting of 28×28 gray-scale images, where each pixel takes a value between 0 and 255. Suppose these gray-scale values are divided by 255 so they are normalized to lie in the interval $[0, 1]$. Then $\mathcal{X} = [0, 1]^{784}$, which is very high dimensional! But, the sampling density $p_X(x)$ is not uniform. On the contrary, looking back at Figure 6, one sees that the MNIST digit images are highly structured. This implies that $p_X(x)$ is (essentially) supported on a much lower dimensional set contained within Q. This is what makes learning possible, but the challenge is that we must take advantage of this fact without being able to precisely know what the supporting set of $p_X(x)$ is.

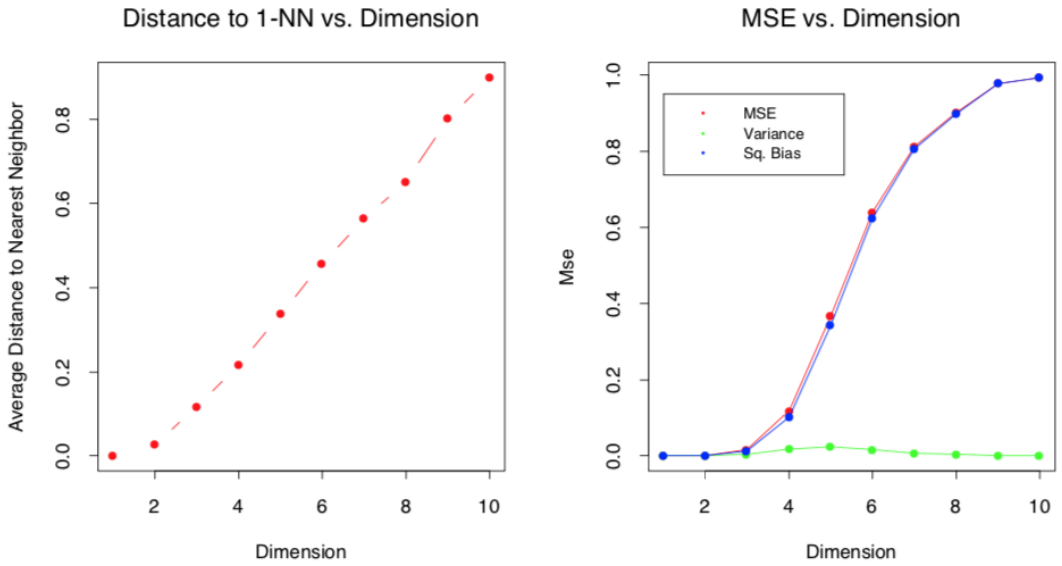


Figure 19: *Plots illustrating Example 4.8. Left: The average distance of the 1-nearest neighbor to the origin over many draws of the training set with $N = 1000$ training points, as a function of the dimension d. Right: The expected test error at the origin for the label function $y = e^{-8\|x\|_2^2}$ (MSE), as a function of the dimension, and broken down into its bias squared component and its variance component. Figure taken from [6].*

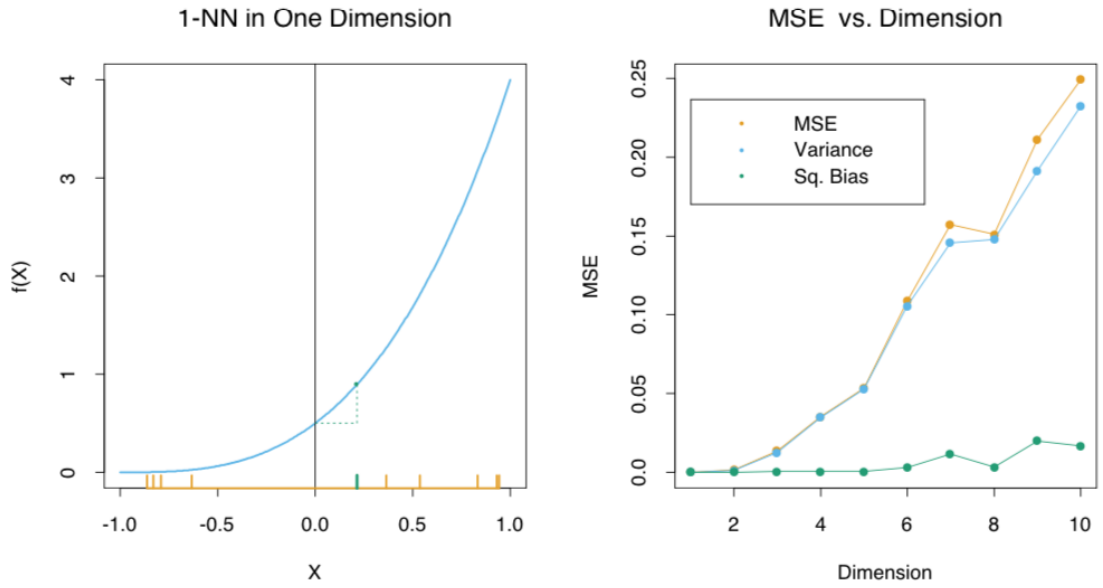


Figure 20: *Additional plots illustrating Example 4.8. In this figure the label function is* $y = (1/2) \cdot (x(1) + 1)^3$*, which only depends on the first dimension of the data point* x*. The variance rapidly increases with the dimension, though, leading to another manifestation of the curse of dimensionality. Figure taken from [6].*

References

[1] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations*, 2015.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition*, 2009.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105. 2012.

[4] Larry Greenemeier. AI versus AI: Self-taught AlphaGo Zero vanquishes its predecessor. *Scientific American*, October 18, 2017.

[5] Pankaj Mehta, Marin Bukov, Ching-Hao Wang, Alexandre G.R. Day, Clint Richardson, Charles K. Fisher, and David J. Schwab. A high-bias, low-variance introduction to Machine Learning for physicists. arXiv:1803.08823, 2018.

[6] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer-Verlag New York, 2nd edition, 2009.

[7] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive Computation and Machine Learning. The MIT Press, 2002.