About the Curse of Dimensionality

Extracted from a Vincent Spruyt blog post

Full article available at http://www.datasciencecentral.com/profiles/blogs/about-the-curse-of-dimensionality

Introduction

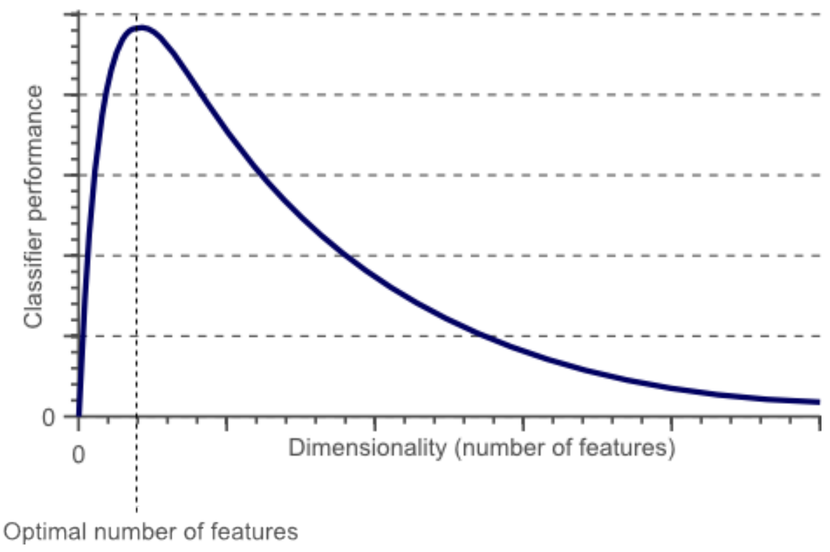
In this article, we will discuss the so-called 'Curse of Dimensionality', and explain why it is important when designing a classifier. In the following sections I will provide an intuitive explanation of this concept, illustrated by a clear example of overfitting due to the curse of dimensionality.

Consider an example in which we have a set of images, each of which depicts either a cat or a dog. We would like to create a classifier that is able to distinguish dogs from cats automatically. To do so, we first need to think about a descriptor for each object class that can be expressed by numbers, such that a mathematical algorithm, i.e. a classifier, can use these numbers to recognize the object. We could for instance argue that cats and dogs generally differ in color. A possible descriptor that discriminates these two classes could then consist of three number; the average red color, the average green color and the average blue color of the image under consideration. A simple linear classifier for instance, could combine these features linearly to decide on the class label:

*If 0.5*red + 0.3*green + 0.2*blue > 0.6 : return cat;*
else return dog;

However, these three color-describing numbers, called features, will obviously not suffice to obtain a perfect classification. Therefore, we could decide to add some features that describe the texture of the image, for instance by calculating the average edge or gradient intensity in both the X and Y direction. We now have 5 features that, in combination, could possibly be used by a classification algorithm to distinguish cats from dogs.

To obtain an even more accurate classification, we could add more features, based on color or texture histograms, statistical moments, etc. Maybe we can obtain a perfect classification by carefully defining a few hundred of these features? The answer to this question might sound a bit counter-intuitive: *no we cannot!* In fact, after a certain point, increasing the dimensionality of the problem by adding new features would actually degrade the performance of our classifier. This is illustrated by the following figure, and is often referred to as 'The Curse of Dimensionality'.



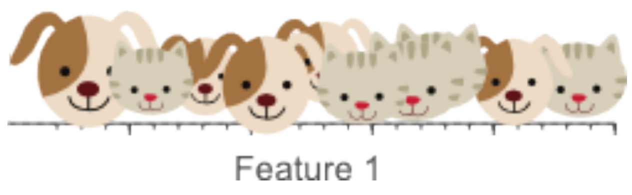
As the dimensionality increases, the classifier's performance increases until the optimal number of features is reached. Further increasing the dimensionality without increasing the number of training samples results in a decrease in classifier performance.

In the next sections we will review why the above is true, and how the curse of dimensionality can be avoided.

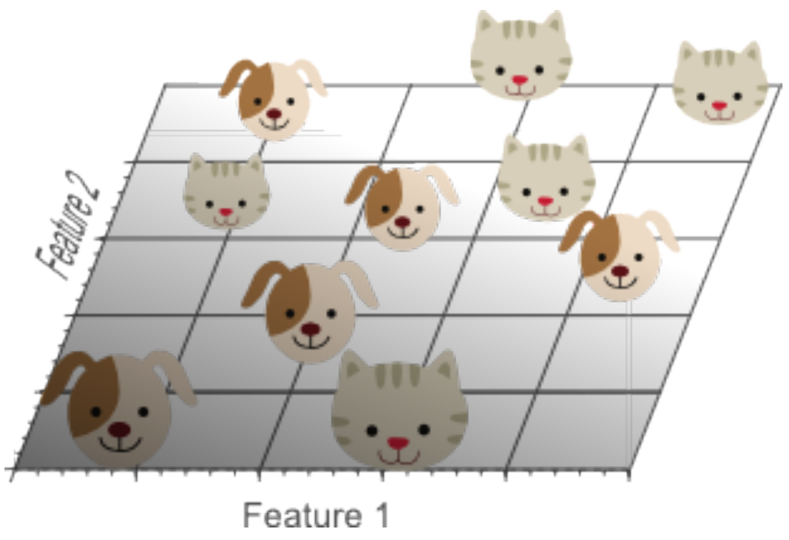
The curse of dimensionality and overfitting

In the earlier introduced example of cats and dogs, let's assume there are an infinite number of cats and dogs living on our planet. However, due to our limited time and processing power, we were only able to obtain 10 pictures of cats and dogs. The end-goal in classification is then to train a classifier based on these 10 training instances, that is able to correctly classify the infinite number of dog and cat instances that we do not have any information about.

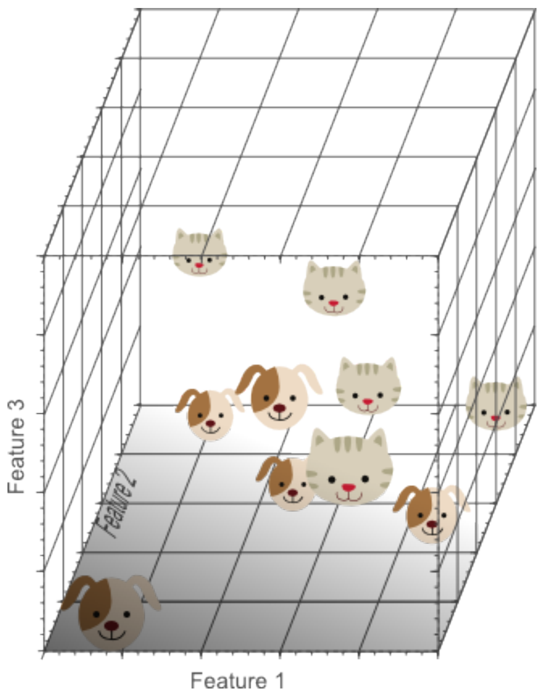
Now let's use a simple linear classifier and try to obtain a perfect classification. We can start by a single feature, e.g. the average 'red' color in the image:



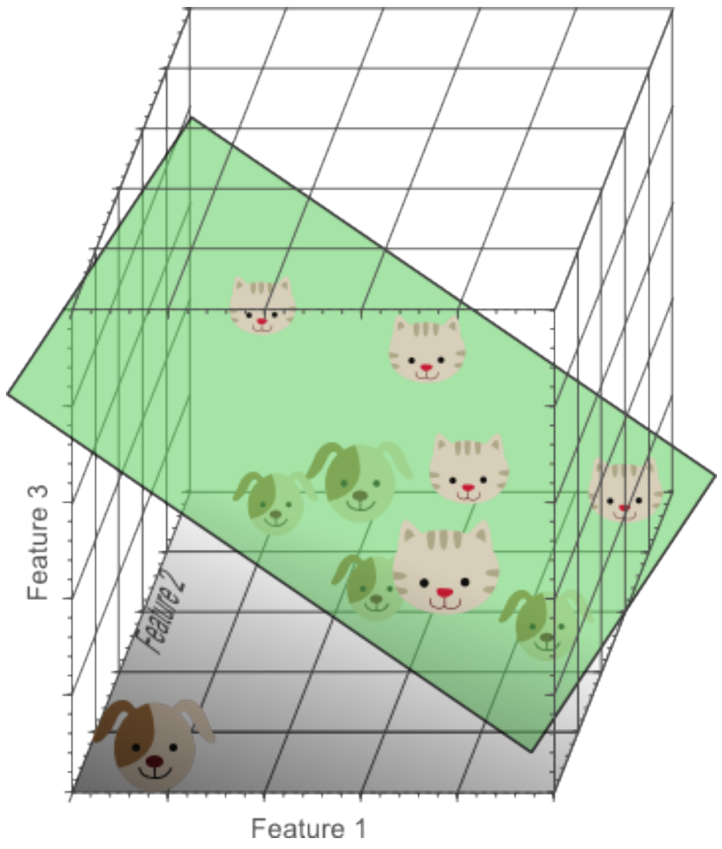
The above figure shows that we do not obtain a perfect classification result if only a single feature is used. Therefore, we might decide to add another feature, e.g. the average 'green' color in the image:



Adding a second feature still does not result in a linearly separable classification problem: No single line can separate all cats from all dogs in this example. Finally we decide to add a third feature, e.g. the average 'blue' color in the image, yielding a three-dimensional feature space:



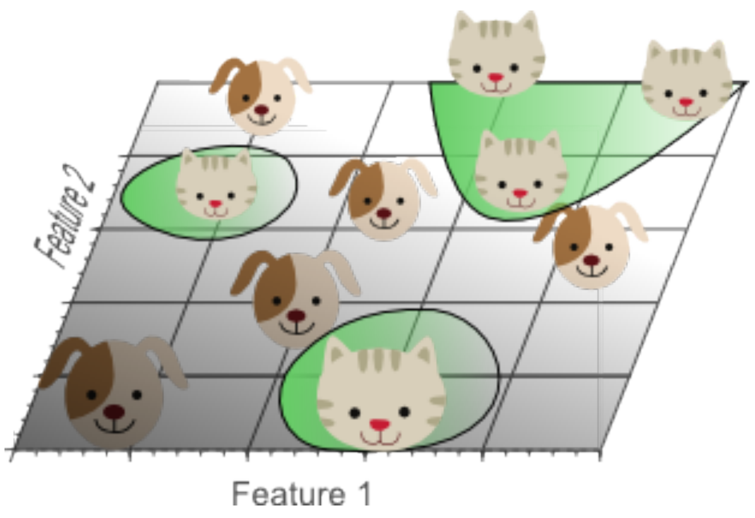
In the three-dimensional feature space, we can now find a plane that perfectly separates dogs from cats. This means that a linear combination of the three features can be used to obtain perfect classification results on our training data of 10 images:



The above illustrations might seem to suggest that increasing the number of features until perfect classification results are obtained is the best way to train a classifier, whereas in the introduction, we argued that this is not the case. However, note how the density of the training samples decreased exponentially when we increased the dimensionality of the problem.

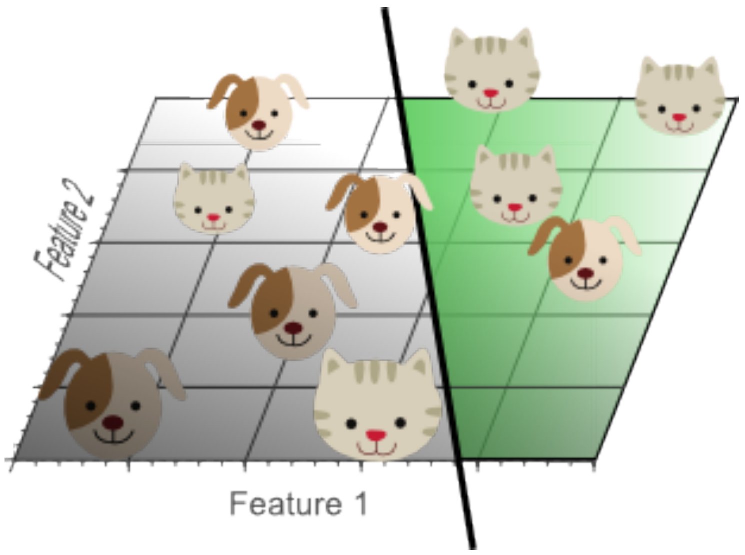
In the 1D case, 10 training instances covered the complete 1D feature space, the width of which was 5 unit intervals. Therefore, in the 1D case, the sample density was 10/5=2 samples/interval. In the 2D case however, we still had 10 training instances at our disposal, which now cover a 2D feature space with an area of 5x5=25 unit squares. Therefore, in the 2D case, the sample density was 10/25 = 0.4 samples/interval. Finally, in the 3D case, the 10 samples had to cover a feature space volume of 5x5x5=125 unit cubes. Therefore, in the 3D case, the sample density was 10/125 = 0.08 samples/interval.

If we would keep adding features, the dimensionality of the feature space grows, and becomes sparser and sparser. Due to this sparsity, it becomes much more easy to find a separable hyperplane because the likelihood that a training sample lies on the wrong side of the best hyperplane becomes infinitely small when the number of features becomes infinitely large. However, if we project the highly dimensional classification result back to a lower dimensional space, a serious problem associated with this approach becomes evident:



The above figure shows the 3D classification results, projected onto a 2D feature space. Whereas the data was linearly separable in the 3D space, this is not the case in a lower dimensional feature space. In fact, adding the third dimension to obtain perfect classification results simply corresponds to using a complicated non-linear classifier in the lower dimensional feature space. As a result, the classifier learns the appearance of specific instances and exceptions of our training dataset. Because of this, the resulting classifier would fail on real-world data, consisting of an infinite amount of unseen cats and dogs that often do not adhere to these exceptions.

This concept is called overfitting and is a direct result of the curse of dimensionality. The following figure shows the result of a linear classifier that has been trained using only 2 features instead of 3:



Although this simple linear classifier seems to perform worse than the non-linear classifier illustrated above, the linear classifier generalizes much better to unseen data because it did not learn specific exceptions that were only in our training data by coincidence. In other words, by using fewer features, the curse of dimensionality was avoided such that the classifier did not overfit the training data.