Data science for Engineers

Prof. Ragunathan Rengasamy
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture - 46
K - Nearest Neighbors (kNN)

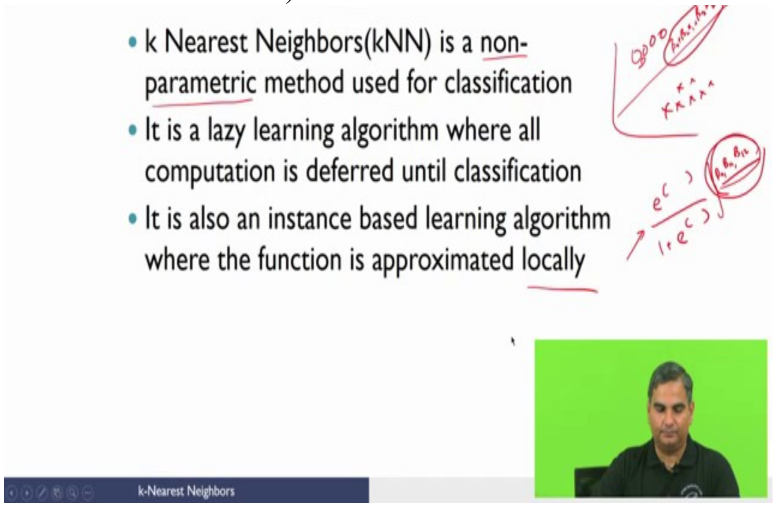
(Refer Slide Time: 00:12)

k-Nearest Neighbors (kNN)

k-Nearest Neighbors 1

In this lecture, we will look at a very very simple yet powerful classification algorithm called the k nearest neighbors. So, let me introduce a k nearest neighbor classification algorithm.

(Refer Slide Time: 00:30)



- k Nearest Neighbors(kNN) is a non-parametric method used for classification
- It is a lazy learning algorithm where all computation is deferred until classification
- It is also an instance based learning algorithm where the function is approximated locally

It is, what is called a nonparametric algorithm for classification. So, let me explain what this non parametric means. Remember when we looked at logistics regression for example, we said, let us say there is data like this and we are going to use the trained data, to develop a hyperplane of this form $\beta_0 + \beta_{11} X_1 + \beta_{12} X_2$ in the 2 d case. And any time a new data point comes, what we do is, we use the parameters that have been estimated from the train data to make predictions about test data.

So, remember we had this e power this term divided by 1 + e power this term right here. So, this function is actually a function of β_0 , β_{11} and β_{12}. So, these are all parameters that have already been derived out of this data. And any time a new test data comes in, it is sent through this function and then you make a prediction. So, this is a parametric method, because parameters have been derived from the data. k nearest neighbor is a different idea, where I do not get parameters like this out of the data. I am going to use the data itself to make classifications. So, that is an interesting different idea that one uses in k nearest neighbors.

I just want to make sure that we get the terminology right. We will later see that the k nearest neighbor, there is one parameter that we use for classifying, which is the number of neighbors that I am going to look at. So, I do not want you to wonder, since we are using anyway a parameter in this k nearest neighbor why am I calling it nonparametric. So, the distinction here is subtle, but I want you to remember this. The number of neighbors that we use in the k nearest neighbor algorithm that you will see later, is actually a tuning parameter for the algorithm, that is not a parameter that I have derived from the data.

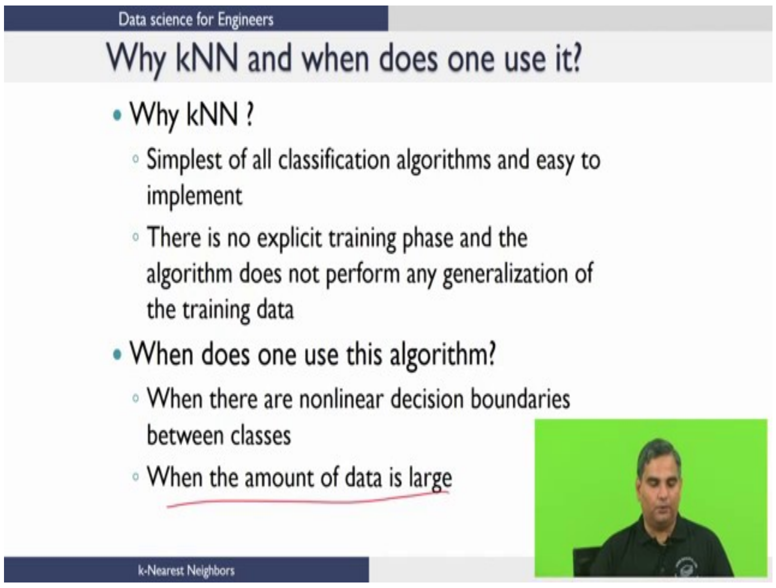
Whereas, in logistics regression, these are parameters I can derive only from the data. I cannot say what these values will be a priori.

Whereas, I could say, I will use a k nearest neighbor with two neighbors three neighbors and so on, so that is a tuning parameter. So, I want you to remember the distinction between a tuning parameter and parameters that are derived from the data and the fact that k nearest neighbor is a nonparametric method, speaks to the fact that we are not deriving any parameters from the data itself. However, we are free to use tuning parameters for k nearest neighbors, so that is an important thing to remember. It is also called a lazy learning algorithm, where all the computation is deferred until classification.

So, what we mean by this is the following. If I give trained data for example, for logistics regression. I have to do this work to get these parameters, before I can do any classification for a test data point . So, without these parameters I can never classify test data points. However, in k nearest neighbor it just give me data and a test data point I will classify.

So, we will see how that is done, but no work needs to be done before I am able to classify a test data point. So, that is an other important difference between k nearest neighbor and logistic regression for example. It is also called as an instant based learning where the function is approximated locally. So, we will come back to this notion of local as I describe this algorithm.

(Refer Slide Time: 04:37)


Data science for Engineers

Why kNN and when does one use it?

- Why kNN ?
  - Simplest of all classification algorithms and easy to implement
  - There is no explicit training phase and the algorithm does not perform any generalization of the training data
- When does one use this algorithm?
  - When there are nonlinear decision boundaries between classes
  - When the amount of data is large

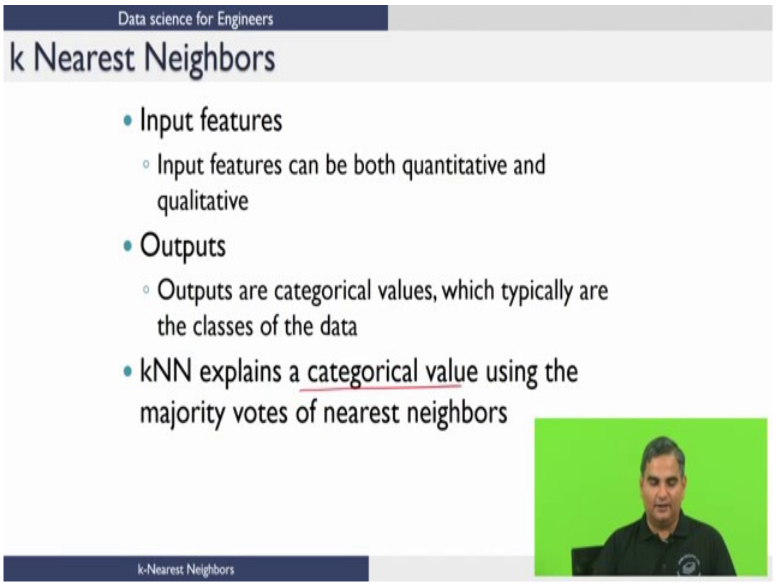
k-Nearest Neighbors

Now we may ask, when do we use this. As I started this lecture I mentioned it simplest of classification algorithms, very easy to implement and you will see when I explain the algorithm how simple it is. There is no explicit training phase and so on and there is no generalization for the training data and all that. It is just that I give the data and then I just wait till they give me a new data point, to say what class it should belong to. Of course, based on the algorithm itself I can also predict for the train data points itself what class they should belong to and then maybe compare it with the label that the data point has and so on.

Nonetheless I am not going to explicitly get some parameters out. And when does one use this algorithm, this is a simple algorithm when there are complicated non-linear decision boundaries, this algorithm actually works surprisingly well, and when you have large amount of data and the train phase can be bogged down by large number of data in terms of an optimization algorithm and so on then you can use this. However, a caveat is you will see as we describe this algorithm when you have more and more data, the classification of nearest neighbor itself, will become complicated.

So, there are ways to address this, but when we say, when the amount of data is large, all that we are saying is since there is no explicit training phase, there is no optimization with a large number of data points, to be able to identify parameters that are useless at later in classification. So, in other words, in other algorithms you will do all the e ort a priori and once you have the parameters then classification becomes, on the test data point becomes, easier. However, since kNN is a lazy algorithm all the, all the calculations are deferred till you had actually have to do something, at that point there might be lot more classic, lot more calculations if the data is large.

(Refer Slide Time: 06:54)


Data science for Engineers

k Nearest Neighbors

- Input features
  - Input features can be both quantitative and qualitative
- Outputs
  - Outputs are categorical values, which typically are the classes of the data
- kNN explains a categorical value using the majority votes of nearest neighbors

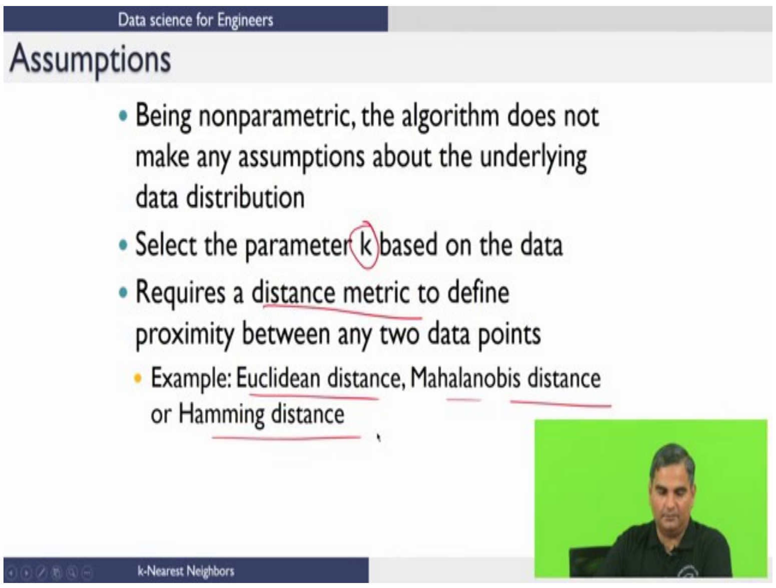
k-Nearest Neighbors

So, the input features for k nearest neighbors could be both quantitative and qualitative, and output are typically categorical values which are what type of class does this data belong to. Now it is not necessary that we use k nearest neighbor only for classification though that is where its used the most. You could also use it with very simple extensions or simple definitions for function approximation problems also, and you will see as I describe this algorithm how it could be adapted for function approximation problems quite easily.

Nonetheless as far as this lecture is concerned, we are going to say the outputs or categorical values, which basically says different classes and what class does this data point belong to. In one word if you were to explain k nearest neighbor algorithm, you would simply say k nearest neighbor explains the categorical value, using the majority votes of nearest neighbors.

So, what basically we are saying is, if there is a particular data point and I want to find out which class this data point belongs to, all I need to do is look at all the neighboring data points and then find which class they belong to and then take a majority vote and that is what is the class that is assigned to this data point. So, its something like if you want to know a person, you know his neighbors, something like that is what use using k nearest neighbors.

(Refer Slide Time: 08:36)



Data science for Engineers

Assumptions

- Being nonparametric, the algorithm does not make any assumptions about the underlying data distribution
- Select the parameter k based on the data
- Requires a distance metric to define proximity between any two data points
    - Example: Euclidean distance, Mahalanobis distance or Hamming distance

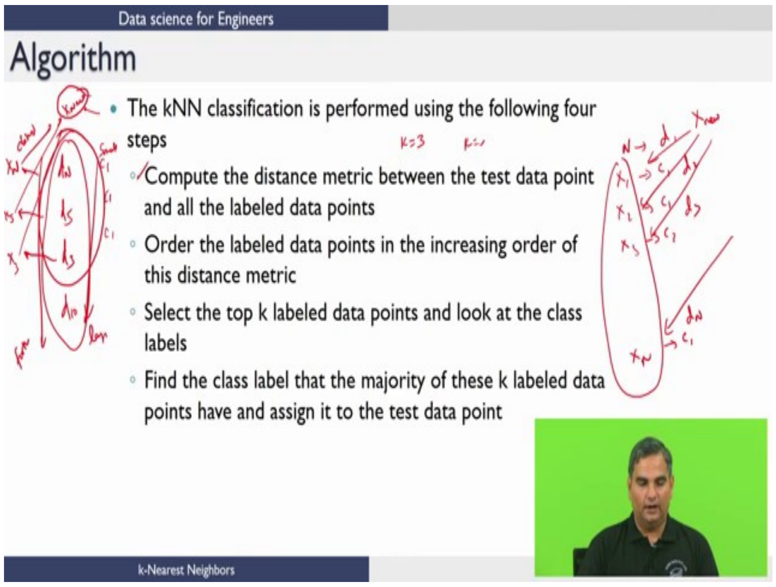
k-Nearest Neighbors

Now remember at the beginning of this portion of data science algorithms I talked about the assumptions that are made by different algorithms. Here for example, because this is a nonparametric algorithm, we really do not make any assumptions about the underlying data distribution. We are just going to look at the nearest neighbors and then come up with an answer. So, we are not going to assume probability distribution or any other linear separability assumptions and so on.

As I mentioned before, this k, the number of neighbors we are going to look at, is a tuning parameter and this is something that you select. So, you use a tuning parameter, run your algorithm and you get good results, then keep that parameter if not you kind of play around with it and then find the best k for your data. The key thing is that because we keep talking about neighbors, and from a data science viewpoint whenever we talk about neighbors, we have to talk about a distance between a data point and its neighbor.

We really need a distance metric for this algorithm to work and this distance metric would basically say what is the proximity between any two data points. The distance metric could be Euclidean distance, Mahalanabis distance, Hamming distance and so on. So, there are several distance metrics that you could use to basically use k nearest neighbor.

(Refer Slide Time: 10:14)



Data science for Engineers

Algorithm

- The kNN classification is performed using the following four steps
  - Compute the distance metric between the test data point and all the labeled data points
  - Order the labeled data points in the increasing order of this distance metric
  - Select the top k labeled data points and look at the class labels
  - Find the class label that the majority of these k labeled data points have and assign it to the test data point

k-Nearest Neighbors

So, in terms of the algorithm itself it is performed using the following four steps. Nothing is done till the algorithm gets a data point to be classified. Once you get a data point to be classified, let us say I have N data points in my database and each has a class label. So for example, X_1 belongs to class 1, X_2 belongs to class 1, X_3 belongs to class 2 and so on, X_n belongs to let us say class 1. So, this is you know a binary situation, binary classification problem. This need not be a binary classification problem; for example, X_2could belong to class 2 and so on.

So, there might be many classes. So, multi class problems are also very very easy to solve using kNN algorithm. So, let us anyway stick to binary problem. Then what you are going to do is, let us say I have a new test point which I call it X_{new} and I want to find out how I classify this. So, the very first step which is what we talk about here, is we find a distance between this new test point and each of the labelled data points in the data set. So for example, there could be a distance d_l between Xv and X_1, d 2 between Xv and X_2, d_3 and so on and d_l. So, once you calculate this distance, then what you do is you have n distances and this is the reason why we said you need a distance metric in last slide for a kNN to work.

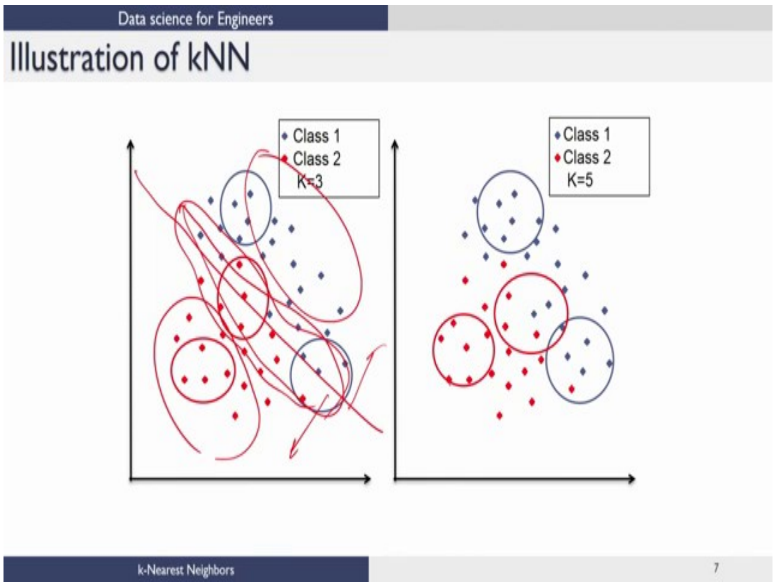
Once we have this distance then what we do, is basically we look at all of these distances and then say, I order the distances from the smallest to the largest. So, let us say if d_n is the smallest distance, so d_n maybe d 5, d 3, d 10, whatever it is, so I order them. This is the smallest to the largest. You can also think of this as closest to the farthest, because the distances are all from X_{new}.

So, the distance is zero then the point is Xv itself. So, any small distance is the closest to X_{new} and as you go down it is further and further. Now the next step is very simple. If let us say you are looking at k nearest neighbors with k = 3, then what you are going to do, is you are going to find the first three distances in this and this distance is from Xn, this distance is from X_5 and this distance is from X_3.

So, once we order this according to distance and go from the smallest to largest, once we sort it in this fashion, then we also know what the corresponding data points are. So, this belongs, this is the data point Xn, so this is the distance between Xn and X_{new}, distance between X_5 and this. So, now, I have these three data points that I picked out from the data set. Now if I want to classify this all that I do is the following, I find out what class these data points belong to. So, if all of them belong to class 1, then I say X_{new} is class 1.If all of them belong to class 2, I say X_{new} is class 2. If two of them belong to class 1 and the third one belongs to class 2, I do a majority vote and still say its class 1. If two of them belong to class 2 and one belongs to class 1 I say its class 2 that is it, that is all the algorithm is.

So, it says to find the class label that the majority of this k label data points have and I assign it to the test data point, very simple. Now I also said this algorithm with minor modifications can be used for function approximation. So, for example, if you so choose to, you could take this and then let us say if you want to predict what an output will be for a new point, you could find the output value for these three points and take an average. For example, very trivial, and then say that is the output corresponding to this, so that becomes of adaptation of this for function approximation problems and so on. Nonetheless for classification this is the basic idea. Now if you said k = 5 then what you do is, you go down to 5 numbers and then do the majority vote, so that is all we do. So, let us look at this very simple idea here.

(Refer Slide Time: 15:05)



Data science for Engineers

Illustration of kNN

k-Nearest Neighbors

7

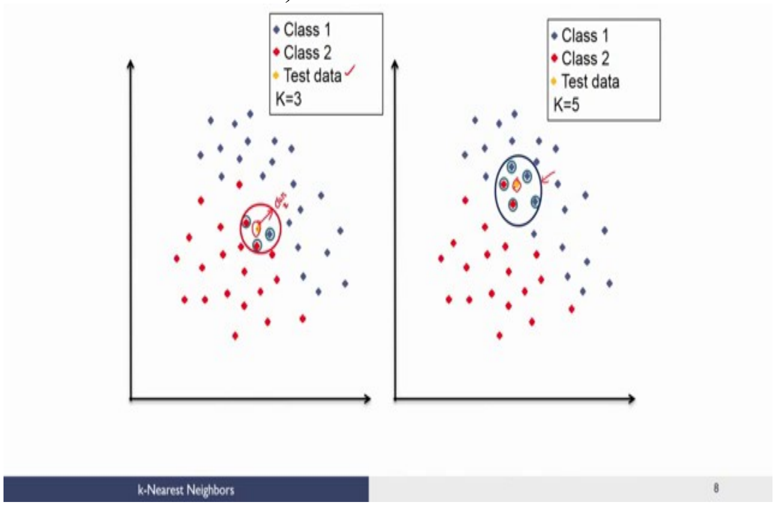
Let us say this is actually the training data itself and then I want to look at k = 3 and then see what labels will be for the training data itself. The blue are actually labeled, so this is supervised. So, the blue are all belonging to class 1 and the red is all belonging to class 2, and then let us say for example, I want to figure out this point here blue point. Though I know the label is blue, what class would k nearest neighbor algorithm say this point belongs to. Say if I want to take k = 3, then basically I have to find three nearest points which are these three, so this is what is represented.

And since the majority is blue, this will be blue. So, basically if you think about it, this point would be classified correctly and so on. Now even in the training set for example, if you take this red point, I know the label is red; how-ever, if I were to run k nearest neighbor with three data points, when you find the three closest point, they all belong to blue. So, this would be misclassified as blue even in the training data set. So, you will notice one general principle is, there is a possibility of data points getting misclassified, only in kind of this region where there is a mix of both of these data points.

However, as you go further away, the chance of miss classification keeps coming down. So, again in some sense you see a parallel, saying if I were to draw a line here and then say this is all red class, this is all blue class. Then points around this line is where the problem is, as they go further away the problems are less. Nonetheless notice how we have never defined a boundary line or a curve here at all, the data points themselves tell you how this boundary is drawn. So, in that sense, its, while it is simple it is also in something sophisticated, because we never have to guess a boundary, the data points themselves define a boundary in some sense. So, I can actually effectively use this algorithm for complicated non-linear boundaries, which you would have to guess a priori if we were using a parametric approach, so that is a key idea here. a similar illustration for K = 5.

Now if I want to let us say a check this data point from the training set itself, then I look at its five neighbors, closest 1 2 3 4 5, all of them are red. So, this is classified as red and so on. So, this is the basic idea.

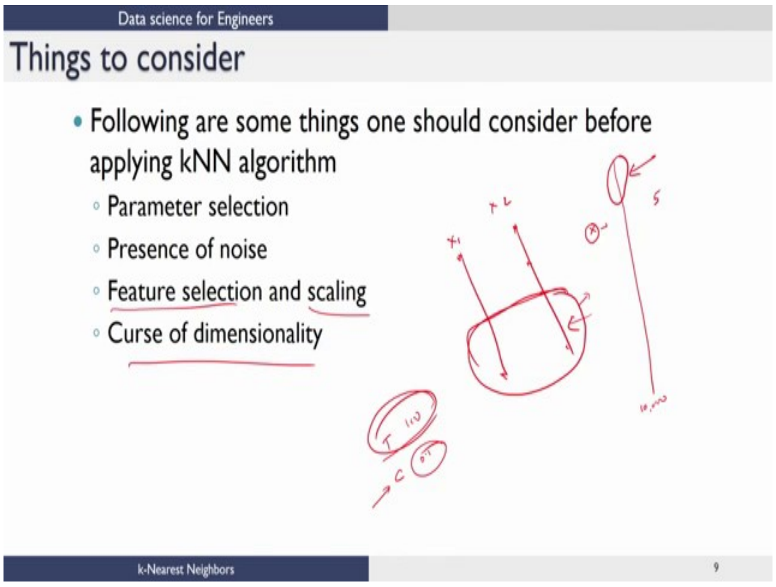
(Refer Slide Time: 18:06)



Now, you do not have to do anything till you get a data point. So, you could verify how well the algorithm will do on the training set itself. However, if you give me a new test data here, so which is what you shown by this data point. Then if you want to do a classification there is no label for this. Remember the other red and blue data points already have a label from prior knowledge, this does not have a label.

So, I want to find out a label for it. So, if I were to use k = 3, then for this data point I will find the three closest neighbors, they happen to be these three data points. Then I will notice that two out of these are red, so this point will get a label + 2. If on the other hand the test data point is here and you were using K = 5 then, you look at the 5 closest neighbor to this point and then you see that two of them are class 2 and three are class 1, so majority voting, this will be put into class 1. So, you will get a label of class 1 for this data point. So, this is the basic idea of k nearest neighbor, so very very simple.

(Refer Slide Time: 19:17)



Data science for Engineers

Things to consider

- Following are some things one should consider before applying kNN algorithm
  - Parameter selection
  - Presence of noise
  - Feature selection and scaling
  - Curse of dimensionality

k-Nearest Neighbors

9

However, these are some of these the things to consider before applying kNN. So, one has to choose the number of neighbors that one is going to use, the value of k, whether its 3 5 7 9 whatever that is, and the results can quite significantly depend on the parameter that you choose. Particularly when you have noise in the data and that has to be taken into account. The other thing to keep in mind when using kNN is that, when you do a distance between two data points X 1 and X 2 let us say, and let us say there are n components in this, the distance metric will take all of these components into picture.

So, since we are comparing distance, then that basically means every at-tribute for this data point and this data point we are comparing distances. The problem with this is that, if for example, there are a whole lot of attributes which actually are not at all important from a classification viewpoint, then what happens is, though they are not important from a classification viewpoint, they contribute in the distance measure. So, there is, there is a possibility of this features actually kind of spoiling the results in k nearest neighbor.

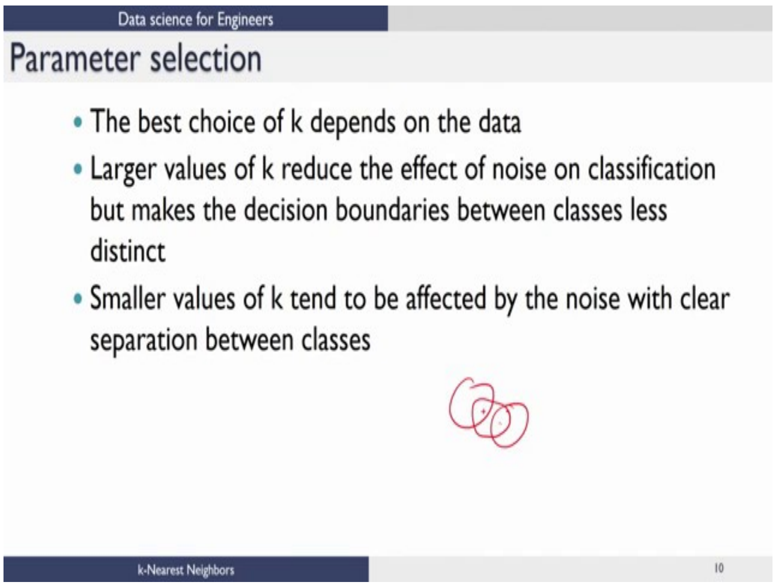
So, it is important to pick features which are which are of relevance in some sense, so the distance metric actually uses only features which will give it the discriminating capacity. So, that is one thing to keep in mind. The other the problem is, these are these are handle able, these are rather easily handled, but these are things to keep in mind when you look at these kinds of algorithms and also particularly this being the first course on data science I am assuming for most of you, these are kinds of things that you might not have thought about before.

So, its worthwhile to kind of think about this, do some mental experiments to see why these kinds of things might be important and so on. Now the other aspect is scaling. So, for example, if there are two attributes, let us say in data temperature and concentration, and temperatures are in values of 100, concentrations are in values of 0.1 0.2 and so on. When you take a distance measure and these numbers will dominate over this.

So, it is always a good idea to scale your data in some format before doing this distance. Otherwise while this might be an important variable from a classification viewpoint, it will never show up, because these numbers are bigger and they will simply dominate the small number. So, feature selection and scaling are things to keep in mind. And the last thing is curse of dimensionality. So, I told you that while this is a very nice algorithm to apply, because there is not much computation that is done at the beginning itself. However, if you notice, if I get a test data point and I have to find let us say the 5 closest neighbor, there is no way in which I can do this, it looks like, unless I calculate all the distances.

So, that can become a serious problem, if the number of data points in my database is very large. Let us say I have 10000 data points, and let us assume that I have an algorithm k nearest neighbor algorithm with K = 5. So, really what I am looking for, is finding 5 closest data points from this data base to this data point. However, it looks like I have to calculate all the 10,000 distances and then sort them and then pick the top 5. So, in other words to get this top 5 have to do so much work. So, there must be smarter ways of doing it, but nonetheless one has to remember the number of data points and number of features, one has to think how to apply this algorithm carefully.

(Refer Slide Time: 24:00)



Data science for Engineers

Parameter selection

- The best choice of k depends on the data
- Larger values of k reduce the effect of noise on classification but makes the decision boundaries between classes less distinct
- Smaller values of k tend to be affected by the noise with clear separation between classes

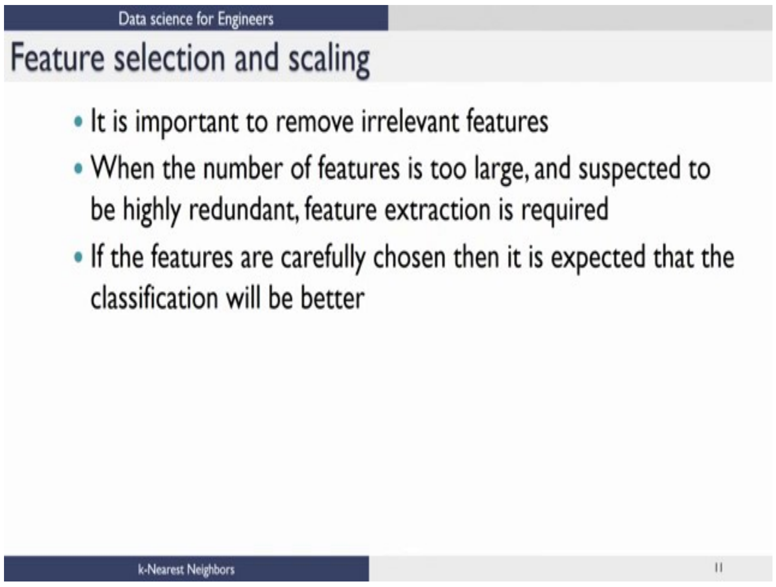
k-Nearest Neighbors

10

So, the best choice of k depends on the data and one general rule of thumb is, if you use large values for k, then clearly you can see you are taking lot more neighbors, so you are getting lot more information. So, the effect of noise on classification can become less. However, if you take large number of neighbors, then your decision boundaries are likely to become less crisp and more di use.

So, because if let us say there are two classes like this, then for this data point if you take a large number of neighbors, then you might pick many neighbors from the other class also, so that can make the boundaries less crisp and more di use. On the fifth slide, flipside, if you use smaller values of k then your algorithm is likely to be affected by noise and outliers, however, your decision boundaries as a rule of thumb are likely to become crisper. So, this is, these are some things to keep in mind.

(Refer Slide Time: 25:08)



Data science for Engineers

Feature selection and scaling

- It is important to remove irrelevant features
- When the number of features is too large, and suspected to be highly redundant, feature extraction is required
- If the features are carefully chosen then it is expected that the classification will be better

k-Nearest Neighbors 11

And as I mentioned before it is important to remove irrelevant features and scaling is also an important idea. So, if you choose your features carefully, then you would get better classification with kNN. So, with this we come to an end of this lecture on k nearest neighbors and following this lecture there will be a case study, which will use k nearest neighbor that will be taught by one of the teaching assistants. And after that I will teach a lecture on k means clustering. Thank you and I look forward to seeing you again in a future lecture.

Thanks.