Chapter 8
k - Nearest Neighbors Algorithm

Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/ labels or categories. Classification predictive modeling is the task of approximating a mapping function (f) from input variables (X) to discrete output variables (y).

For example, spam detection in email service providers can be identified as a classification problem. This is s binary classification since there are only 2 classes as spam and not spam. A classifier utilizes some training data to understand how given input variables relate to the class. In this case, known spam and non-spam emails have to be used as the training data. When the classifier is trained accurately, it can be used to detect an unknown email.

Classification belongs to the category of supervised learning where the targets also provided with the input data. There are many applications in classification in many domains such as in credit approval, medical diagnosis, target marketing etc.

8.1 Type of Learners

There are two types of learners in classification as Eager learners and Instance-based learning (lazy learners).

Eager Learners

Eager learners construct a classification model based on the given training data before receiving data for classification. It must be able to commit to a single hypothesis that covers the entire instance space. Due to the model construction, eager learners take a long time for training and less time to predict.

Ex. Decision Tree, Naive Bayes, Artificial Neural Networks

Instance-based Learning (Lazy Learners)

Lazy learners simply store the training data and wait until a testing data appear. When it does, classification is conducted based on the most related data in the stored training data. Compared to eager learners, lazy learners have less training time but more time in predicting.

Ex. k-nearest neighbor, Case-based reasoning

8.2 k-Nearest Neighbors

The k-nearest neighbors (kNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems. However, it is more widely used in classification problems in the industry.

"kNN which stand for K Nearest Neighbor is a Supervised Machine Learning algorithm that classifies a new data point into the target class, depending on the features of its neighboring data points."

8.2.1 Features of kNN Algorithm

The kNN algorithm has the following features:

- kNN is a Supervised Learning algorithm that uses labeled input data set to predict the output of the data points.
- It is one of the simplest Machine learning algorithms and it can be easily implemented for a varied set of problems.
- It is mainly based on feature similarity. kNN checks how similar a data point is to its neighbor and classifies the data point into the class it is most similar to.
- Unlike most algorithms, kNN is a non-parametric model which means that it does not make any assumptions about the data set. This makes the algorithm more effective since it can handle realistic data.
- kNN is a lazy algorithm, this means that it memorizes the training data set instead of learning a discriminative function from the training data.
- kNN can be used for solving both classification and regression problems.

8.2.2 The kNN Algorithm

The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other. K-nearest neighbors (kNN) algorithm uses 'feature similarity' to predict the values of new datapoints which further means that the new data point will be assigned a value based on how closely it matches the points in the training set. We can understand its working with the help of following steps:

Step 1 – For implementing any algorithm, we need dataset. So during the first step of kNN, we must load the training as well as test data.

Step 2 – Next, we need to choose the value of K i.e. the nearest data points. K can be any integer.

Step 3 – For each point in the test data do the following –

- Calculate the distance between test data and each row of training data with the help of any of the method namely: Euclidean, Manhattan or Hamming distance. The most commonly used method to calculate distance is Euclidean.
- Now, based on the distance value, sort them in ascending order.
- Next, it will choose the top K rows from the sorted array.
- Now, it will assign a class to the test point based on most frequent class of these rows.

Step 4 – End

The following is an example to understand the concept of K and working of kNN algorithm:
Suppose we have a dataset which can be plotted in Fig 8.1.

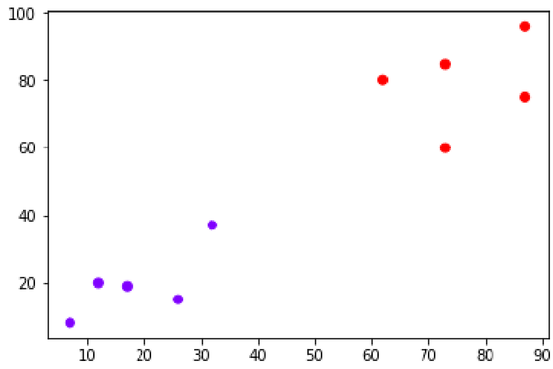


Fig. 8.1

Now, we need to classify new data point with black dot (at point 60,60) into blue or red class. We are assuming K = 3 i.e. it would find three nearest data points. It is shown in Fig 8.2.

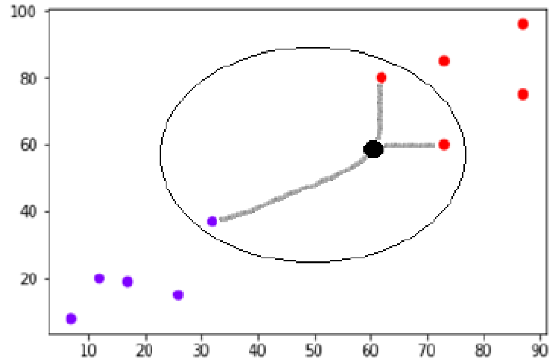
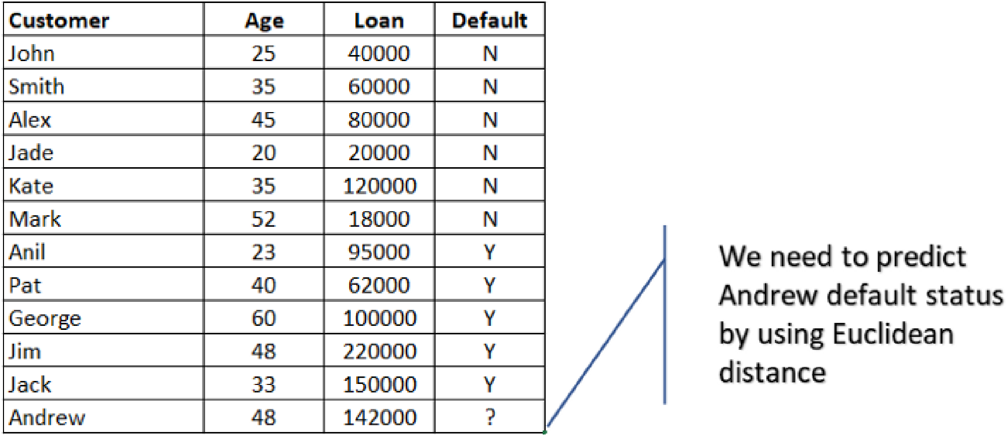


Fig. 8.2

We can see in the above diagram the three nearest neighbors of the data point with black dot. Among those three, two of them lies in Red class hence the black dot will also be assigned in red class.

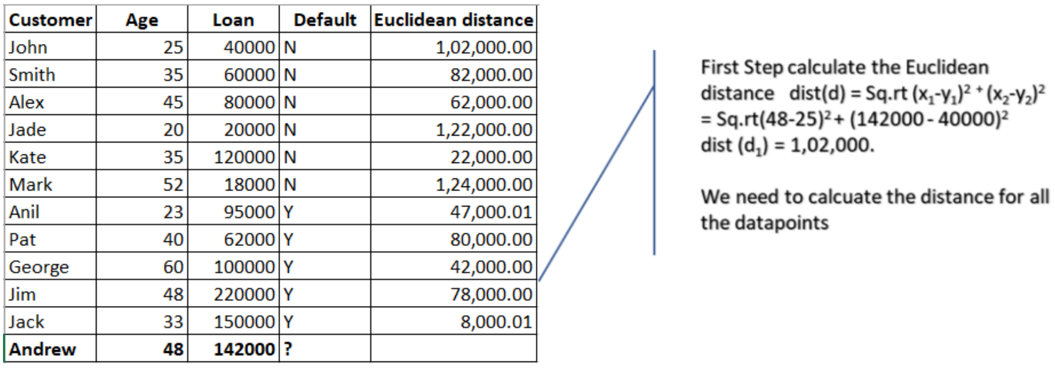
kNN: Example



Customer	Age	Loan	Default
John	25	40000	N
Smith	35	60000	N
Alex	45	80000	N
Jade	20	20000	N
Kate	35	120000	N
Mark	52	18000	N
Anil	23	95000	Y
Pat	40	62000	Y
George	60	100000	Y
Jim	48	220000	Y
Jack	33	150000	Y
Andrew	48	142000	?

We need to predict Andrew default status by using Euclidean distance

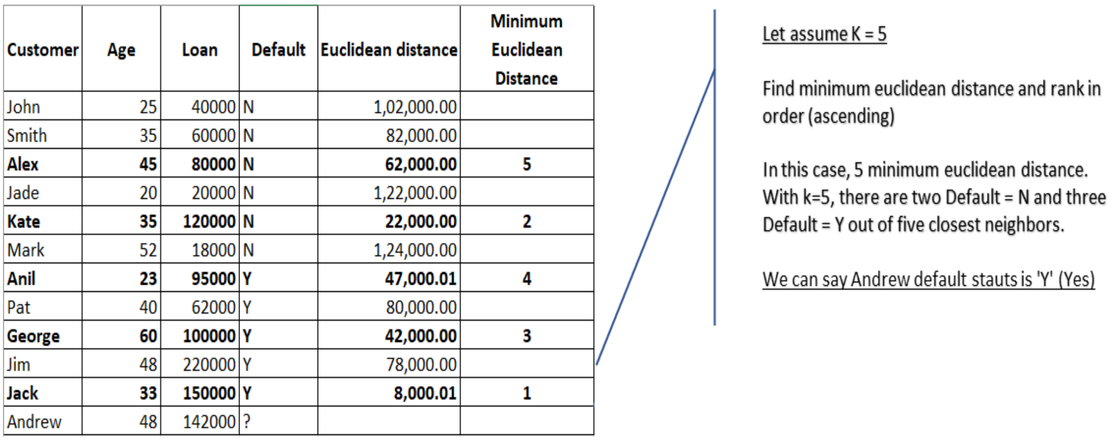
Calculate Euclidean distance for all the data points.



Customer	Age	Loan	Default	Euclidean distance
John	25	40000	N	1,02,000.00
Smith	35	60000	N	82,000.00
Alex	45	80000	N	62,000.00
Jade	20	20000	N	1,22,000.00
Kate	35	120000	N	22,000.00
Mark	52	18000	N	1,24,000.00
Anil	23	95000	Y	47,000.01
Pat	40	62000	Y	80,000.00
George	60	100000	Y	42,000.00
Jim	48	220000	Y	78,000.00
Jack	33	150000	Y	8,000.01
Andrew	**48**	**142000**	**?**	

First Step calculate the Euclidean distance dist(d) = Sq.rt $(x_1-y_1)^2 + (x_2-y_2)^2$ = Sq.rt$(48-25)^2 + (142000 - 40000)^2$ dist (d_1) = 1,02,000.

We need to calcuate the distance for all the datapoints



Customer	Age	Loan	Default	Euclidean distance	Minimum Euclidean Distance
John	25	40000	N	1,02,000.00	
Smith	35	60000	N	82,000.00	
Alex	**45**	**80000**	**N**	**62,000.00**	**5**
Jade	20	20000	N	1,22,000.00	
Kate	**35**	**120000**	**N**	**22,000.00**	**2**
Mark	52	18000	N	1,24,000.00	
Anil	**23**	**95000**	**Y**	**47,000.01**	**4**
Pat	40	62000	Y	80,000.00	
George	**60**	**100000**	**Y**	**42,000.00**	**3**
Jim	48	220000	Y	78,000.00	
Jack	**33**	**150000**	**Y**	**8,000.01**	**1**
Andrew	48	142000	?		

Let assume K = 5

Find minimum euclidean distance and rank in order (ascending)

In this case, 5 minimum euclidean distance. With k=5, there are two Default = N and three Default = Y out of five closest neighbors.

We can say Andrew default stauts is 'Y' (Yes)

With K=5, there are two Default=N and three Default=Y out of five closest neighbors. We can say default status for Andrew is 'Y' based on the major similarity of 3 points out of 5.

8.2.3 How do we choose the factor K?

First let us try to understand what exactly does K influence in the algorithm. Different K could have different results as shown in the figure 8.3.

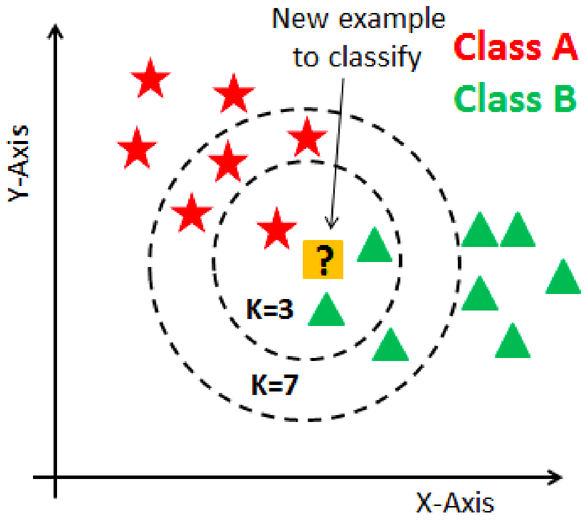


Fig. 8.3

In the following fig. 8.4, we have two classes of data, namely class A (squares) and Class B (triangles). The problem statement is to assign the new input data point to one of the two classes by using the KNN algorithm. The first step in the KNN algorithm is to define the value of 'K'. But what does the 'K' in the KNN algorithm stand for? 'K' stands for the number of Nearest Neighbors and hence the name K Nearest Neighbors (KNN).

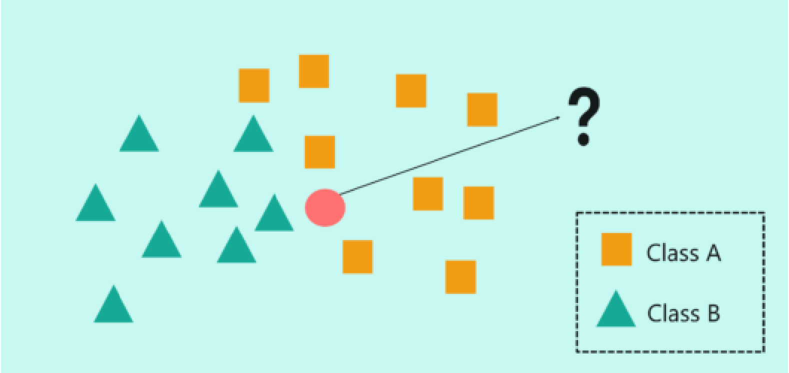


Fig. 8.4

If we consider the value of 'K' as 3. This means that the algorithm will consider the three neighbors that are the closest to the new data point in order to decide the class of this new data point. The closeness between the data points is calculated by using the distance measures such as Euclidean and Manhattan distance. At 'K' = 3, the neighbors include two squares and 1 triangle. So, if I were to classify the new data point based on 'K' = 3, then it would be assigned to Class A (squares) as shown in figure 8.5.

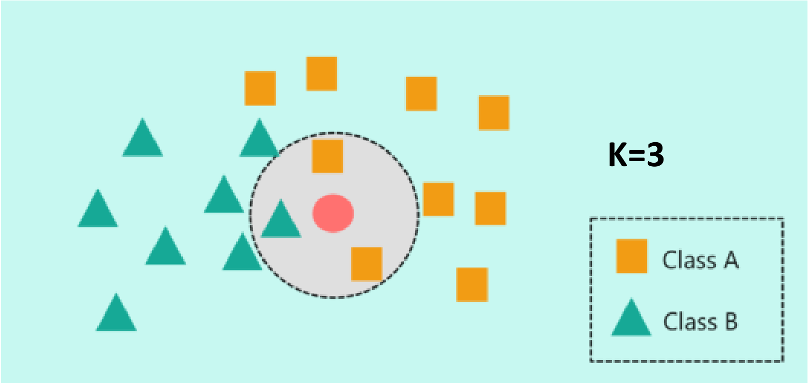


Fig. 8.5 if K=3

But what if the 'K' value is set to 7? So the algorithm will look for the seven nearest neighbors and classify the new data point into the class it is most similar to. At 'K' = 7, the neighbors include three squares and four triangles. So, if we classify the new data point based on 'K' = 7, then it would be assigned to Class B (triangles) since the majority of its neighbors were of class B.

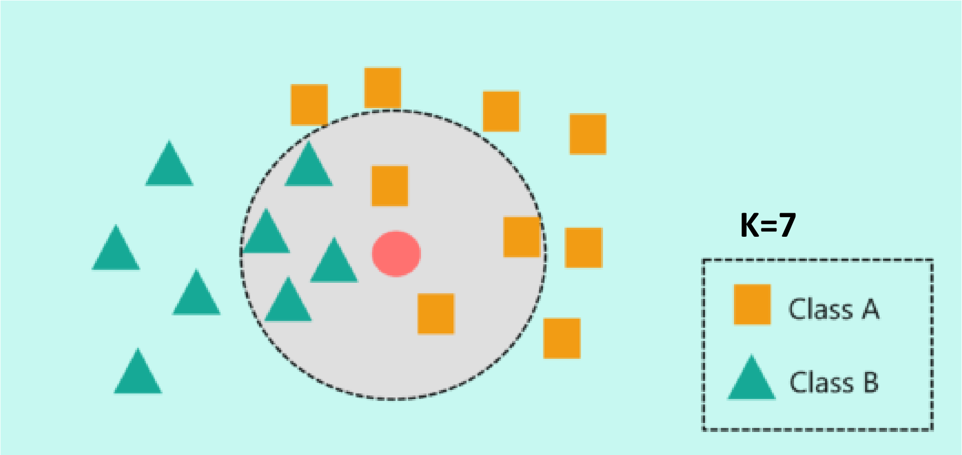


Fig. 8.6 if K=7

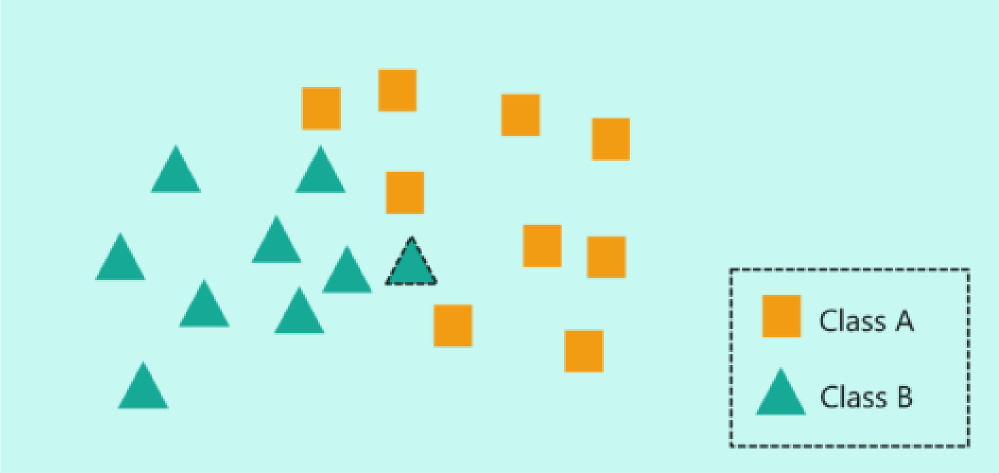


Fig. 8.7 Final Classification

The value of k can be determined experimentally.

- Start with k=1 and use a test set to validate the error rate of the classifier
  Repeat with k=k+2
- Choose the value of k for which the error rate is minimum

Note: Try and keep the value of k odd in order to avoid confusion between two classes of data

8.2.4 Continuous-Valued Target Functions

In nearest-neighbor learning the target function may be either discrete-valued or real valued. If k-NN approximating continuous-valued target functions then Calculate the mean value of the k nearest training examples rather than calculate their most common value.

$$f : \Re^d \rightarrow \Re \qquad \hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} f(x_i)}{k}$$

8.2.4 Distance Weighted kNN

Refinement to kNN is to weight the contribution of each k neighbor according to the distance to the query point x_q

- Greater weight to closer neighbors

For discrete target functions

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\arg\max} \sum_{i=1}^{k} w_i \delta(v, f(x_i))$$

$$w_i = \begin{cases} \dfrac{1}{d(x_q, x_i)^2} & if \quad x_q \neq x_i \\ 1 & else \end{cases}$$

For real valued functions

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} w_i f(x_i)}{\sum_{i=1}^{k} w_i}$$

$$w_i = \begin{cases} \dfrac{1}{d(x_q, x_i)^2} & if \quad x_q \neq x_i \\ 1 & else \end{cases}$$

8.2.5 Advantage and disadvantage of k - Nearest Neighbor Algorithm

Pros:

- No assumptions about data distribution, useful in real world application
- Simple algorithm to explain and understand
- It can use for both classification and regression

Cons:

- Computationally expensive, because the algorithm stores all of the training data
- High memory requirement, again, it stores all of the training data
- Prediction stage might be slow (with big N)

8.3. Choice of distance metric: Another parameter to be set by the user of the k nearest neighbors rule is the metric according to which distance are measured. The choice of metric plays an important role in the performance of the nearest neighbor classifier for a given sample size n. Different distance metrics can be used when calculating distance between the object points.

Minkowski distance: It can be universally described as Minkowski metric which is given in Eq. 1:

$$Dist(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}$$

where, different values of p≥1 result in different commonly-used metric.

Manhattan distance: When p = 1, that is Manhattan distance (Eq. 2). It is the absolute distance total between two points on the standard coordinate system. The correlation of different features is not taken into account here:

$$Dist(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i| \right)$$

Euclidean distance: When p = 2, that is Euclidean distance (Eq. 3). It is the most common metric used in KNN:

$$Dist(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^2 \right)^{\frac{1}{2}}$$

Euclidean distance measures an absolute distance between various points in a multidimensional space. At the same time, Euclidean metric need to ensure dimension indicators in the same level because of distance calculating based on the absolute values of various dimension characteristics. This metric should be used when the different features are not strongly correlated.