KNN

1. Basic concepts, principles, and applications of k nearest neighbor (KNN) algorithm

The KNN algorithm is a basic classification and regression method. In the application of artificial intelligence, its classification usage is mainly considered.

The KNN algorithm is to give a training data set and, for a new input instance, find the K instances closest to the instance in the training data set.

Most of these K instances belong to a certain class, and the input instance is classified into this class. (This is like the minority obeying the majority). As shown in fig 1.

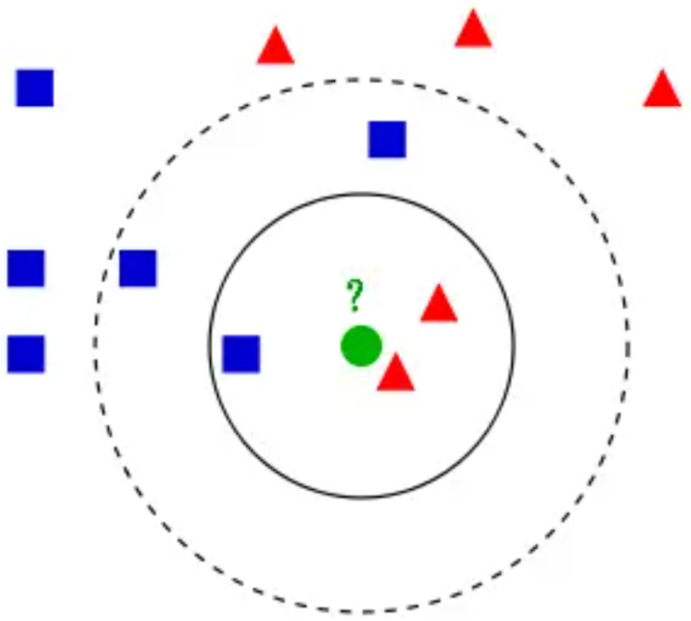

Fig 1. KNN classification diagram

As shown in the figure above, there are two different types of sample data, represented by small blue squares and small red triangles.

The data marked by the green circle in the middle of the picture is the data to be classified.

This is our purpose, here comes a new data point, what is the category I want to get it?

Okay, let's classify the green dots based on the idea of k nearest neighbors:

- If K=3, the three closest points to the green dot are 2 small red triangles and 1 small blue square. It is determined that the green point to be classified belongs to the red triangle category.
- If K=5, the five nearest neighbors of the green dot are 2 red triangles and 3 blue squares. It is determined that the green point to be classified belongs to the blue square category.

2. The selection of k in the KNN algorithm and the importance of feature normalization

If we choose a smaller k value, the model will become complex and prone to overfitting. as shown in figure 2.

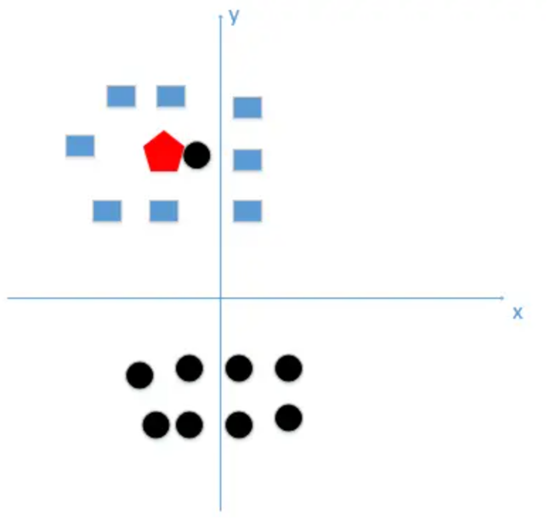

Fig 2. Training data and points to be classified.

There are two categories in the picture above, one is a black dot and the other is a blue rectangle. Now our points to be classified are red pentagons.

Determine which category the points to be classified should be classified according to our KNN algorithm steps.

From Figure 2, we can know that the pentagon is closest to the black dot. If k is equal to 1, it is finally determined that the point to be classified is a black dot.

From this process, we can easily feel that something is wrong. If k is too small, for example equal to 1. Then the model can easily learn the noise, and it can also be easily determined as a noise category.

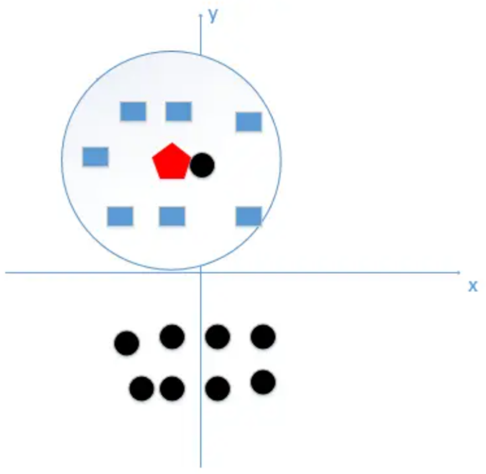

Fig 3. K=8

If k is larger, k is equal to 8. We can easily get that our correct classification should be a blue rectangle, as shown in Figure 3.

The overfitting means that the accuracy on the training set is very high, but the accuracy on the test set is low.

From the above example, we know that too small k will lead to overfitting. It is easy to learn some noise into the model and ignore the true distribution of the data.

If we choose a larger k value, it is equivalent to using training data in a larger neighborhood for prediction. At this time, training instances that are far away from the input instance (not similar) will also affect the prediction, causing the prediction to be wrong.

An increase in the k value means that the overall model becomes simpler.

If k=N (N is the number of training samples), then no matter what the input instance is, it will simply be predicted to belong to the class with the most training instances. As shown in fig 4.

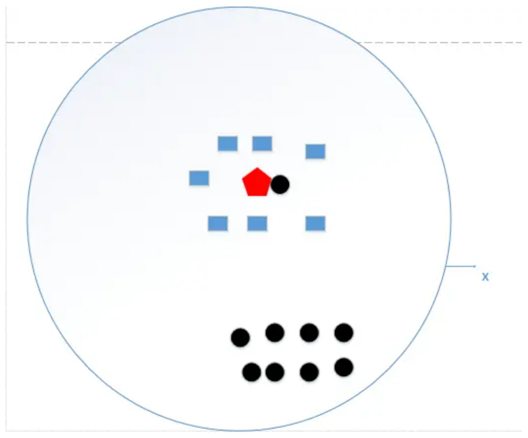

Fig 4. K=N

In the picture, there are 8 black circles and 7 rectangles. If k=N, then the red pentagon belongs to the black circle (obviously wrong).

At this time, the model is too simple and completely ignores a large amount of useful information in the training data instances.

Therefore, the k value can neither be too large nor too small. In this example, the choice of k value, the range between the red circle boundaries in Figure 5 is the best:

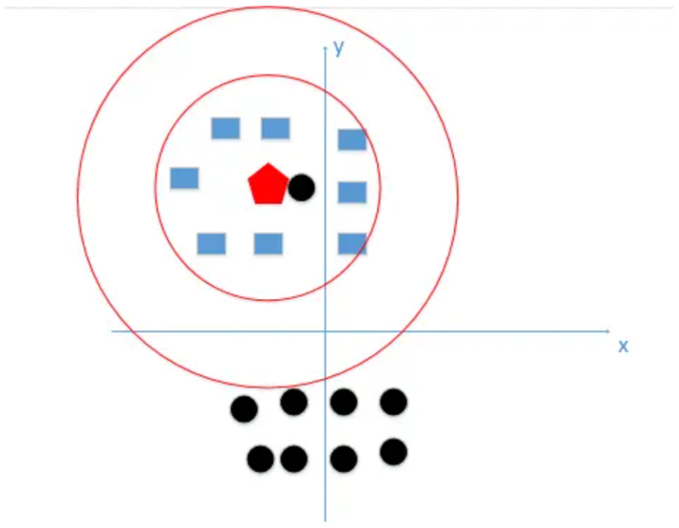

Fig 5. K value selection interval

For the K value, the cross-validation method is generally used to select the optimal k value. (In other words, the key to selecting the k value is experimental parameter adjustment)

3. Application of KNN algorithm in XAI

The application of KNN algorithm in XAI is mainly reflected in its simplicity and intuitiveness. Here are some potential applications of KNN in XAI:

1. Transparency and Interpretability: In an OCR system, KNN can be used to classify characters based on similarity to known examples. For instance, if a handwritten 'A' is classified, KNN can reveal its decision by showing the 'K' most similar handwritten 'A's from the training dataset, making the process transparent.
2. Local Explanations: KNN provides explanations for specific OCR classifications. For example, if a particular handwritten letter is misclassified, examining the nearest neighbors can reveal why the algorithm made that mistake, showing similar but differently classified characters.
3. Feature Importance: In OCR, while KNN itself does not measure feature importance, its decision-making process can give insights. For instance, altering the distance metric to

emphasize different character features (like stroke thickness or slant) can reveal which features are more influential in classifying a particular character.

4. Contrastive Analysis: KNN can be used for contrastive analysis in OCR. By comparing the nearest neighbors of a misclassified character with those correctly classified, one can identify distinguishing features or common errors, such as confusing 'I' with 'l'.
5. Data Visualization: For OCR, visualizing the data with KNN can be insightful, especially for low-dimensional feature representations. Plotting characters and their nearest neighbors can reveal clustering patterns and how different characters are grouped together.
6. Anomaly Detection and Explanation: In OCR, KNN can help detect and explain anomalies. For example, if a character is significantly different from its neighbors (like a uniquely styled letter), it can be flagged as an outlier, prompting further investigation into its distinct features or potential misclassification.
7. Model Comparison and Validation: Comparing KNN with complex OCR models (like Convolutional Neural Networks) can help validate the more complex model's decisions. If KNN's predictions for character recognition vastly differ from those of a deep learning model, it might suggest areas where the complex model's understanding of characters could be improved or is lacking.