Curso de especialización en Inteligencia Artificial y Big Data

UD04. DESARROLLO DE APLICACIONES DE IA: ALGORITMOS

ALGORITMO DE LOS VECINOS MÁS CERCANOS - KNN (RESUMEN)

Carlos Sáenz Adán

1. Algoritmo de los vecinos más cercanos

Antes de comenzar con la explicación, te recomiendo que eches un ojo al siguiente vídeo. En este video se realiza una explicación muy bien ilustrada de la regresión lineal.

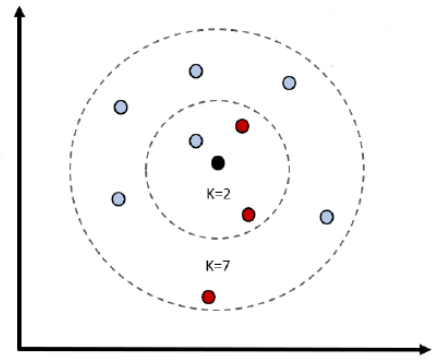
https://www.youtube.com/watch?v=FHHuo7xEeo4

1.1. Introducción matemática

El algoritmo de los vecinos más cercanos (KNN) es uno de los algoritmos de aprendizaje supervisado más sencillos siendo un ejemplo de aprendizaje por analogía.

La idea básica sobre la que se fundamenta este algoritmo es que un nuevo caso se va a clasificar en la clase más frecuente a la que pertenecen sus k vecinos más cercanos. En el caso de utilizar kNN sobre un problema de clasificación, el algoritmo devolverá la clase a la cual debe pertenecer el caso desconocido. Esta elección se fundamentará en el uso del voto (mayoría) para decidir el valor más adecuado, pudiéndose ser el voto ponderado o no. Por otro lado, si el algoritmo se utiliza para resolver problemas de regresión, el algoritmo devolverá la media de los valores de los vecinos.

En la siguiente Figura se observan 9 ejemplos pertenecientes a dos clases distintas, 5 a la clase A en azul y 3 a la clase B en rojo. Cada ejemplo tiene dos atributos que ayudan a clasificarlo, x_1 y x_2. Como se puede observar, la distancia entre los ejemplos propios de cada clase es por lo general menor que la distancia con los ejemplos de la clase contraria, por lo que aparentan agruparse según su clase. Al clasificar un nuevo ejemplo, representado con el punto negro del centro. Si se elige k = 3, el nuevo ejemplo se clasificará con la clase B, dado que 2 de esos 3 ejemplos más cercanos pertenecen a esa clase. Sin embargo, si elegimos k = 6, el ejemplo caerá dentro de la clase A, dado que 4 de los 6 ejemplos pertenecen a esa clase. La k es por lo tanto uno de los parámetros más determinantes del algoritmo



1.2. Hiperparámetros

- **k**: Número de vecinos a considerar en la predicción.
- **radius**: Radio del espacio de parámetros a utilizar para la predicción.
- **weights**: función de ponderación de contribución de los vecinos utilizada en la predicción.
- **distance**: la métrica de distancia a utilizar para determinar los vecinos más cercanos.

1.3. Implementación con scikit-lean

Para generar modelos KNN la clase utilizada es KNeighborsClassifier. La cual admite ciertos parámetros a la hora de ser construida:

- **n_neighbors**: el número de vecinos a considerar en la predicción.
- **weights**: puede ser uniform (todos los vecinos tienen el mismo peso) o distance (el peso es en función de la distancia, cuanto más cerca más peso)
- **algorithm**: puede ser "ball_tree", "kd_tree", "brute" (fuerza bruta) o "auto" (decidirá el más apropiado en base a los argumentos del método fit)
- **p**: sirve para definir la métrica de distancia para determinar los vecinos más cercanos. Si definimos 1, utilizará manhattan_distance, si definimos 2 utilizará distancia euclídea. Por defecto se utilizará la distancia de minkowski.

```
from sklearn.neighbors import KNeighborsClassifier

model = KNeighborsClassifier(n_neighbors=3,
                             weights='uniform',
                             algorithm='ball_tree',
                             p=1)
model.fit(X_train,y_train)
print('Model score: ', model.score(X_test,y_test))
```