KNN en Python

1 INTRODUCCIÓN

El algoritmo a desarrollar es el algoritmo de clasificación supervisada de los K vecinos más cercanos (KNN).

Se trata de un algoritmo de clasificación supervisada ya que, apoyándose en un conjunto de datos en el que cada individuo está etiquetado con una clase, asigna a cada individuo nuevo que llega una clase.

Para ello se utiliza un conjunto de variables predictoras, en concreto en nuestro algoritmo, se analiza la distancia de los valores de las variables del nuevo individuo respecto a cada caso del dataset. Los K casos que estén más cerca del nuevo individuo determinarán su clase.

> → Para el cálculo de la distancia entre casos, utilizaremos el concepto de la distancia euclídea:
>
> $$d(x, y) = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2}$$
>
> Debido a ello, es importante normalizar las variables, puesto que si tenemos una variable que toma unos valores muy grandes respecto a otra, la primera influirá mucho más en el cálculo de la distancia que la segunda.

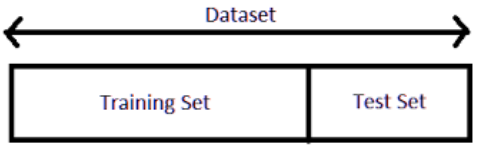
2 EXPERIMENTACIÓN

El problema a resolver se centra en el dataset mtcars, que tiene los datos de diversos modelos de coches como por ejemplo el peso (wt), su potencia en caballos (hp), el número de carburadores (carb), etc. Dadas esas variables predictoras deberemos determinar el consumo del coche en millas por galón.

Para simplificar el problema, en lugar de tener que determinar el número concreto de la métrica del consumo, alteraremos la naturaleza de la variable de discreta a continua, es decir, solo tendremos que indicar si el consumo es alto o bajo (1 o 0).

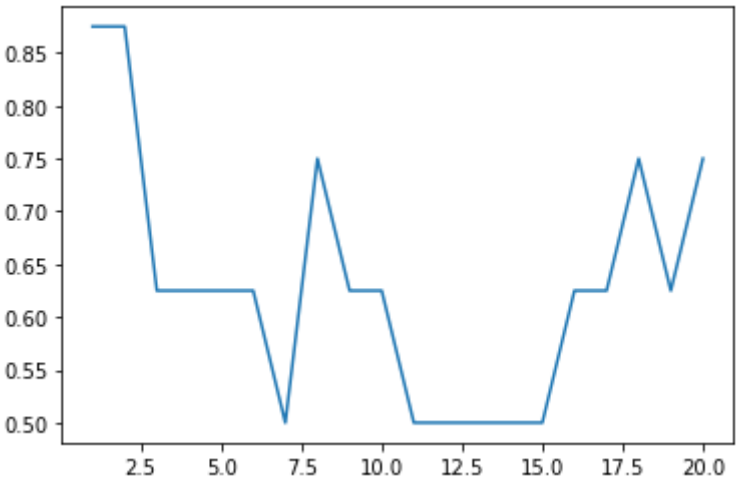
Dado que solo disponemos de un conjunto de entrenamiento, para medir la precisión o veracidad de las clases predecidas por nuestro algoritmo, dividiremos el conjunto en dos particiones (train/test). Una partición será el conjunto de referencia y el otro lo usaremos para probar nuestro algoritmo contra casos de los que no sabemos la clase. Tras ejecutar el algoritmo sobre el conjunto de test, comprobaremos si nuestro algoritmo ha acertado las clases reales.

A este método de evaluación se le llama Holdout, en concreto nosotros usaremos un Holdout con un p=0.66, es decir, nuestra partición de train comprenderá el 66% del dataset original y el de train el 33% restante.

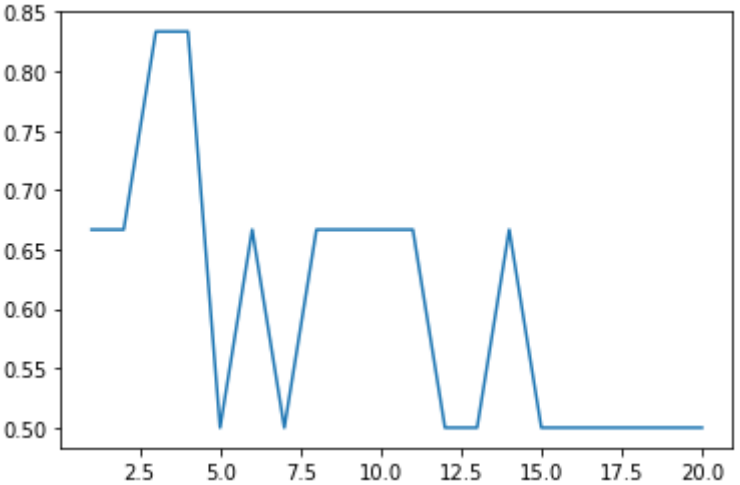


Se ha testeado el modelo para diferentes semillas, cambiando el valor de K en un rango de 1 hasta 20. Estos son algunos resultados:

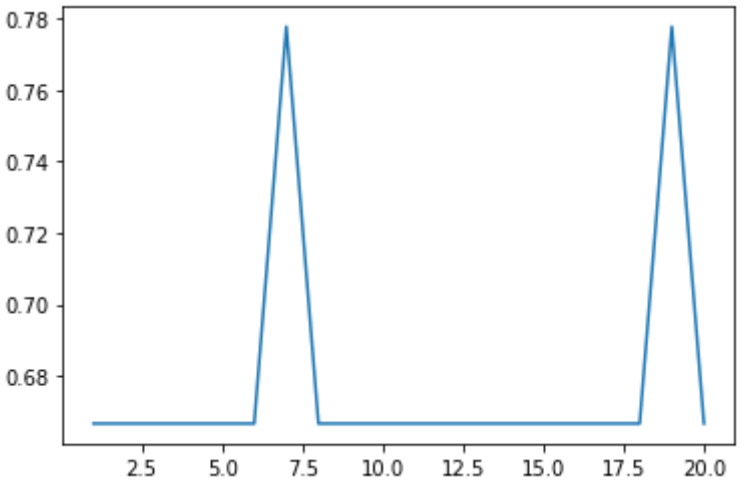
- seed: 7



- seed: 10



- seed: 100



Por los resultados podemos decir que la K óptima suele estar entre el 7 y el 8, aunque como vemos los gráficos varían mucho entre las semillas, debido a que tenemos un dataset pequeño y que además lo estamos dividiendo para aplicar el método holdout explicado anteriormente.

3 CONCLUSIONES

El algoritmo funciona ya que compara los valores de las variables de entrada y en función de los individuos más cercanos determina la clase del nuevo caso.

Parece un razonamiento muy simple pero es muy efectivo, ya que si escogemos variables que sean determinantes para nuestro problema es normal que los individuos de la misma clase tengan un valor parecido para las variables entre sí y que difiere de los valores que tendrán los individuos de otras clases.

Además de la mejora previa del algoritmo como podrían ser decidir que variables son realmente determinantes a la hora de predecir la clase, evitar casos ruidosos, recopilar más datos para entrenar al algoritmo, etcétera; también se pueden aplicar algunas mejoras a nivel del algoritmo.

Existen muchas variables de K-NN como podría ser la del pesado de variables, en esta variante los individuos tienen mayor o menor peso para decidir la clase del individuo en función de a qué distancia del mismo se encuentran.

Como esta existen otras muchas variantes del algoritmo que se podrían probar, es posible que dependiendo del tipo problema una de estas variantes funcione mejor que la otra.

4 OPINIÓN

4.1 ¿Crees que te ha ayudado a aprender cosas nuevas?

Sí, ya que en el resto de prácticas sólo hemos visto cómo se aplican los algoritmos sobre un dataset con diferentes variantes de los mismos y hemos analizado los resultados y por qué se daban de esa forma.

Aunque es verdad que es importante tener una base teórica también es importante ver más a fondo como se codifican los algoritmos a nivel de programación.

4.2 ¿Te ha parecido mucho trabajo?

No, aunque si que es verdad que al principio fue algo costoso ya que aunque los pasos estaban bien explicados y el programa bien modulado es un poco engorroso manipular los dataframes de la librería Pandas de cara a generar los dos subsets y eliminar las etiquetas sin ninguna explicación previa.

4.3 ¿Te parece buena idea invitar a nuevos profes con nuevas ideas?

Sí ya que aportan diferentes formas de encarar el laboratorio y aplicar los conceptos teóricos vistos en clase.