Universidad Nacional de Rosario

Introducción al Aprendizaje Automatizado

Trabajo Práctico IV

Villagra Martín

26 de junio de 2017

Introducción

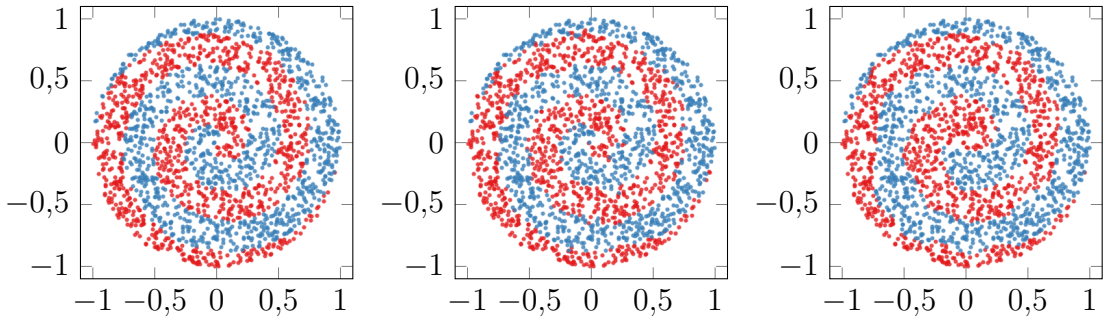
Este trabajo analiza el desempeño de el clasificador de k-vecinos. Se observa su comportamiento bajo distintos datasets y se explican los resultados.

Ejercicio a

Se utilizó el lenguaje Python. La implementación se encuentra en el archivo *knn.py*. En caso de empate se toma una clase al azar.

Ejercicio b

En la Figura 1 se aprecian las predicciones con ambos métodos. Obtuvimos un error de alrededor del 5 %, ligeramente menor que con redes neuronales (7 %). Sin embargo, observando las predicciones vemos que los límites entre las clases no están tan definidos como en redes. Así pues, los errores provienen de estas fronteras confusas, mientras que en redes los errores son por los límites mal colocados.



(a) Conjunto de Test. (b) Predicción de k-vecinos. (c) Predicción con redes neuronales.

Figura 1: Predicciones sobre el conjunto de test del dataset espirales.

Para este caso tomar $K = 1$ es óptimo pues, excepto en los bordes, el punto más cercano va a ser de la misma clase. De esto inferimos que cuando los conjuntos no se superponen, es decir cada clase se presenta forma continua en el espacio de búsqueda y la cantidad de muestras en la frontera es relativamente baja entonces k-NN funciona excelentemente. Cabe destacar que esto rara vez es el caso en el mundo real.

Ejercicio c

Los resultados se resumen en la Figura 2. Se aprecia un buen desempeño especialmente cuando se elige cuidadosamente el K.

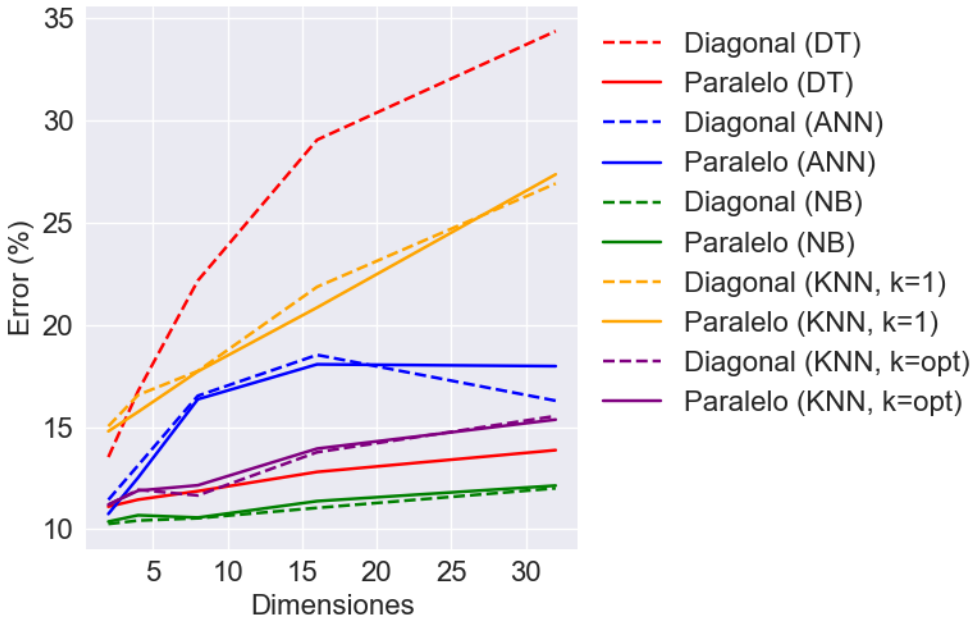


Figura 2: Error en test en función de la cantidad de dimensiones para los diferentes clasificadores.

En cuánto a la elección del K óptimo se observa en la Figura 3 que en promedio a medida que aumentan las dimensiones el K también lo hace. Si se fija $K = 1$ para dimensiones grandes hay errores más altos, como se puede ver en la Figura 2. Esto se atribuye que a medida que aumentan las dimensiones las muestras están cada vez más dispersas y es más probable que el vecino más cercano sea de una clase distinta, especialmente en la frontera. Para contrarrestar esto es por lo que se aumenta K, que produce un efecto suavizador. No se puede apreciar una diferencia notable entre Diagonal y Paralelo en la elección de K. A pesar de esto sigue habiendo un error mayor a Naive-Bayes.

En conclusión podemos afirmar que cuando hay solapamiento o fronteras no definidas entre las clases es necesario ajustar el K para evitar sobreajuste.

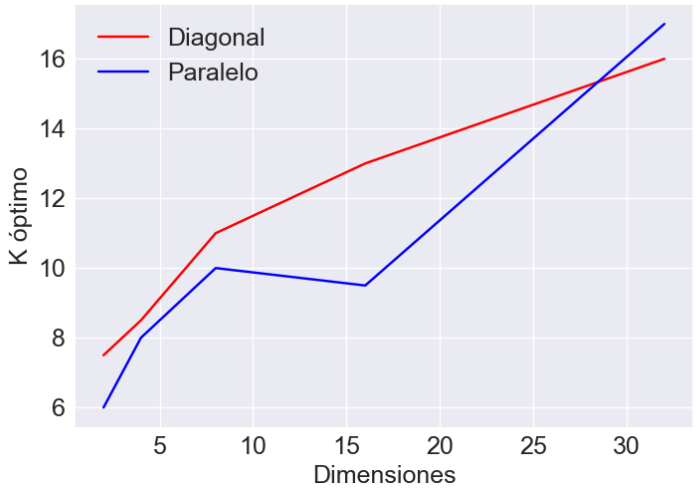


Figura 3: K óptimo en función de la cantidad de dimensiones del dataset.

Ejercicio d

Se utilizó el lenguaje Python. La implementación se encuentra en el archivo *dnn.py*. Ahora en lugar de tener variable K, se tiene que ajustar la distancia D.

En la Figura 4 se aprecia una mejora con esta técnica especialmente para dimensiones altas. Esto puede deberse a que cuando los datos están muy dispersos k-NN tiene en cuenta vecinos que están demasiado alejados del punto a predecir, lo cual no sucede con este método. No hay diferencias significativas entre Paralelo y Diagonal, este método es independiente de la rotación de los datos.

En la Figura 5 se ve como se requieren distancias más grandes para mayores dimensiones. Esto se explica de manera similar a lo que pasaba en el Ejercicio c. Otro detalle interesante es la diferencia entre Paralelo y Diagonal. El D óptimo es aproximadamente $\sqrt{d}$ para Paralelo y d para Diagonal, donde d es la cantidad de dimensiones. Esta relación es explicable pues Diagonal es una rotación y un escalamiento de Paralelo. Específicamente, Diagonal es Paralelo escalado por $\sqrt{d}$. Entonces tiene sentido que la distancia óptima sea escalada también por la misma constante.

Ejercicio e

Una de las mayores debilidades de K-nn ocurre cuando los atributos tienen distintos pesos. Por ejemplo, si estamos prediciendo si personas van a ir a cierto hotel de 5 estrellas entonces el salario de esa persona influye mucho más que otros atributos.

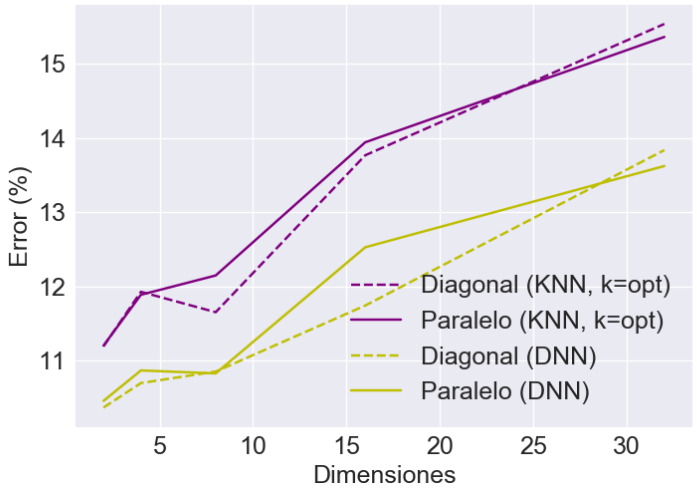


Figura 4: Error en test en función de la cantidad de dimensiones.

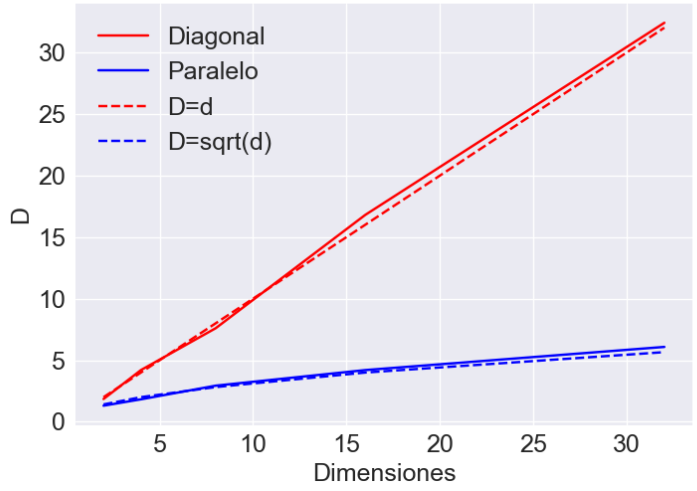


Figura 5: D óptimo en función de la cantidad de dimensiones del dataset.

K-nn trata a todas las dimensiones por igual. Tampoco se pueden modelar relaciones entre los atributos (i.e. si tiene hijos el salario debe ser aún mayor).

Sin embargo si el conjunto de train es una buena muestra del dataset, no se me ocurrió ninguna ejemplo donde haya sobreajuste de la manera que solicita el enunciado.