Fundamentos de Aprendizaje Automático

Práctica 4

Grupo1461
Mario Valdemaro García Roque
Manuel Reyes Sánchez

Apartado 1 – Detalles de la implementación de K-means

Hemos realizo un código muy modular, realizando el algoritmo en alto nivel desarrollando posteriormente cada una de las partes que lo componen.

Tenemos el siguiente código para la parte de entrenamiento:

```
public void entrenamiento(Datos datosTrain) {
      generarCentroides(datosTrain);            //1#

      do{
            asignacionCluster(datosTrain);      //2#
            reubicarCentroides(datosTrain);     //3#

      }while(fin.fin(centroides,clusters));     //#4
      prediccionCentroide(datosTrain);          //#5
}
```

1#: Generamos aleatoriamente los centroides. Para realizar esta operación escogemos aleatoriamente n datos, siendo n el número de clusters que le hemos indicado al algoritmo a la hora de crear el clasificador, siendo la posición donde se encuentra el dato que ha salido elegido el centroide del cluster.

#2: Usando una tabla de tamaño N, donde N es el numero de datosTrain, indicamos para cada dato cual es el indice del centroide mas cercano de acuerdo a la función de distancia que se indico en el constructor.

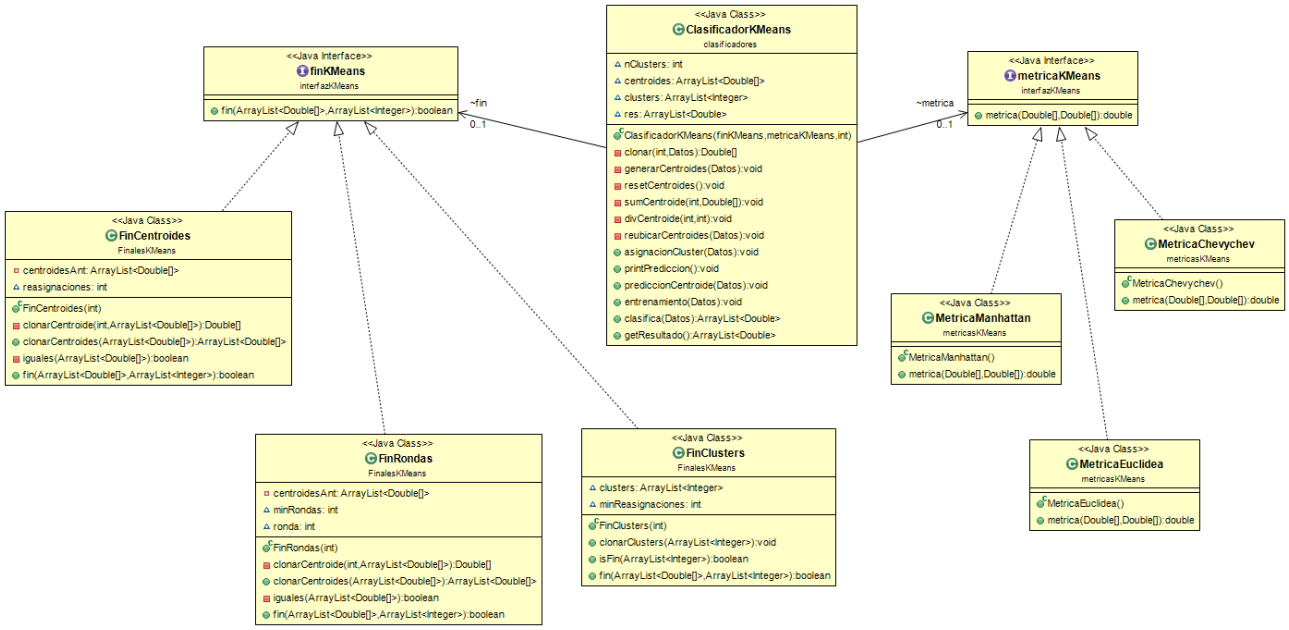
#3: Una vez ya sabemos que datos están en cada cluster reubicamos los centroides de estos.

#4: Comprobamos si se ha producido la condición de parada.

#5: Finalmente una vez tenemos el estado final tenemos que ver que dato predice cada cluster, para ello hacemos la moda de los datos que se encuentran en ese cluster, lo cual es un método rápido y eficaz, pero se podría usar por ejemplo los datos con menor distancia al cluster o los k datos mas cercanos para elegir que dato predecirá el cluster, o incluir peso a cada dato de acuerdo a la distancia al cluster.

Para clasificar datos lo único que hacemos es coger el dato y ver que centroide es el mas cercano al cluster y predecimos su clase.

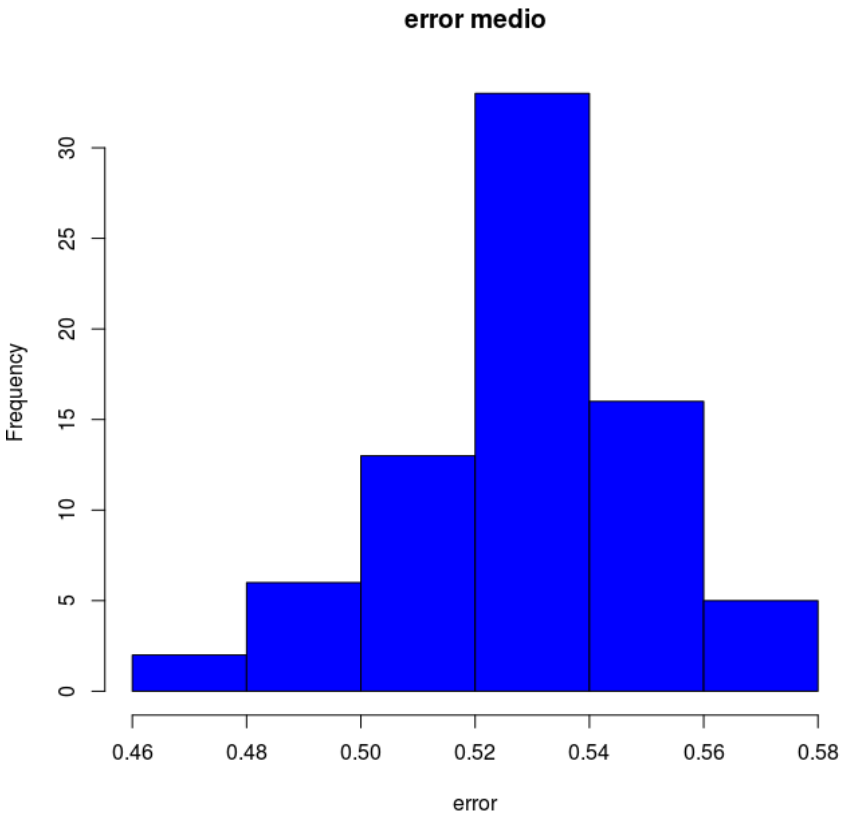
Podemos ver por último el diagrama de clases de la implementación:



Podemos ver como K-Means tiene todas las funciones que el algoritmo necesita y eran llamadas en el algoritmo. Por otro lado hay dos interfaces que sirven para implementar funciones de fin y de distancia. Además podemos ver las implementaciones que hemos realizado de cada una.

Apartado 2 – Evaluación de K-means con K=10

A continuación realizamos una ronda de 300 pruebas siendo el numero de clusters 10, la métrica usada es distancia euclídea y la condición de parada es un mínimo de resignaciones en los clusters. Obteniendo los siguientes errores:



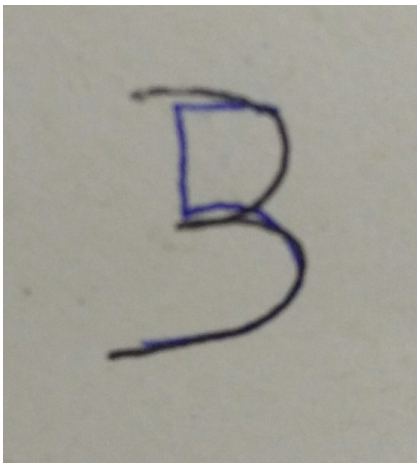
Como se pude ver el error medio esta entorno al 50% por lo que podemos decir que las hipótesis que teníamos al principio de que cada cluster presenta un clase y que una clase entera se va a englobar en un cluster concreto no se acercan mucho a la realidad.

Vamos a ver entonces a que se asigna cada cluster y como es la matriz de confusión, para ello vamos a tomar una muestra de validación cruzada sobre que predice cada cluster y la matriz final de validación cruzada. Obtenemos lo siguiente:

```
Cluster 0 predice:6.0
Cluster 1 predice:7.0
Cluster 2 predice:6.0
Cluster 3 predice:3.0
Cluster 4 predice:0.0
Cluster 5 predice:4.0
Cluster 6 predice:1.0
Cluster 7 predice:6.0
Cluster 8 predice:1.0
Cluster 9 predice:0.0
```

Como ya comentábamos la hipótesis que teníamos de que cada cluster se asignara una clase es falsa. Podemos ver claramente como varios clusters predicen la misma clase y como hay clases que no las predice ningún cluster, esto se puede deber a que primero los centroides iniciales se eligen aleatoriamente, por lo que nada nos asegura que vayan a estar repartidos uniformemente por todas las clases, y a pesar de que se vayan reconfigurando cada época puede que al llegar a la fase final de entrenamiento 2 clusters o mas se engloben a la misma clase.

Podemos intentar entender porque algún numero no tiene clase con un ejemplo concreto, si observamos el ejemplo, no hay cluster asignado a 5, esto puede ser a que ha sido engoblado por el 3.



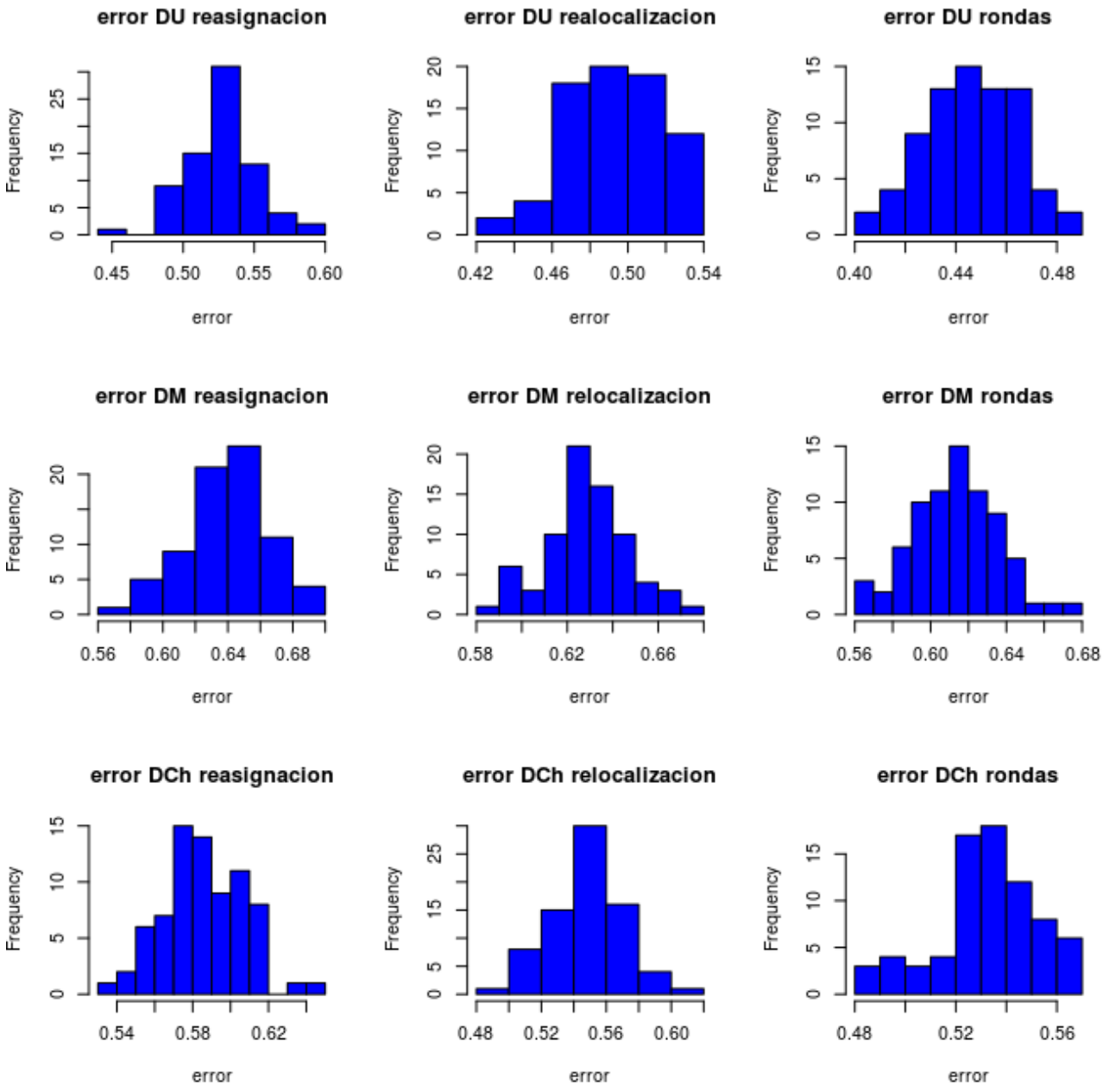
Como vemos toda la mitad inferior puede ser llegar a ser indentica en estos números, y la barra superior también es similar.

Y la siguiente matriz de confusión:

```
Matriz de confusión:
0     1     2     3     4     5     6     7     8     9
------------------------------------------------------------------------------
--
[36]  [0]   [1]   [4]   [0]   [6]   [7]   [1]   [4]   [1]   -->60 | 0
[1]   [25]  [8]   [9]   [7]   [4]   [8]   [2]   [5]   [10]  -->79 | 1
[0]   [4]   [20]  [3]   [0]   [0]   [0]   [7]   [3]   [0]   -->37 | 2
[1]   [4]   [1]   [22]  [1]   [17]  [1]   [2]   [4]   [2]   -->55 | 3
[2]   [1]   [0]   [1]   [31]  [4]   [1]   [2]   [1]   [18]  -->61 | 4
[1]   [3]   [0]   [0]   [0]   [2]   [3]   [1]   [1]   [0]   -->11 | 5
[5]   [5]   [3]   [1]   [2]   [9]   [24]  [0]   [5]   [0]   -->54 | 6
[2]   [3]   [12]  [3]   [2]   [2]   [3]   [26]  [7]   [4]   -->64 | 7
[0]   [0]   [3]   [3]   [0]   [2]   [1]   [1]   [14]  [2]   -->26 | 8
[0]   [3]   [0]   [2]   [5]   [2]   [0]   [6]   [4]   [11]  -->33 | 9
```

Como observamos encontramos bastante error en la matriz de confusión. Algunos datos se predicen mal y otros casi no se predicen como por ejemplo el 5 que se ha precedido solo 11 veces y únicamente 2 de ellas se han acertado. Unas de las muchas cosas curiosas que encontramos es que por ejemplo el 9 y el 4 presentan una confusión entre ellos altísima y tiene sentido ya que son números que se parecen mucho entre ellos al escribirlos. Lo mismo ocurre entre el 2 y el 7.
La matriz de confusión reafirma el ejemplo del 3 y el 5, pues podemos observar que los 5 se clasifican mayoritariamente como 3.

Adicionalmente hemos calculada que métricas junto con que tipo de detección de final de ronda obtiene el mejor error medio, para ello hemos realizado 75 muestras de datos sobre 9 configuraciones distintas, las configuraciones usadas en cada gráfica se puedes observar en el titulo de cada gráfica, pero los parámetros de parada de cada una son, 15 rondas como máximo o que no haya cambios de una ronda a otra, 15 puntos de los cluster cambiados y 50 desplazamientos de centroides sobre todas las ordenadas. A continuación las gráficas:

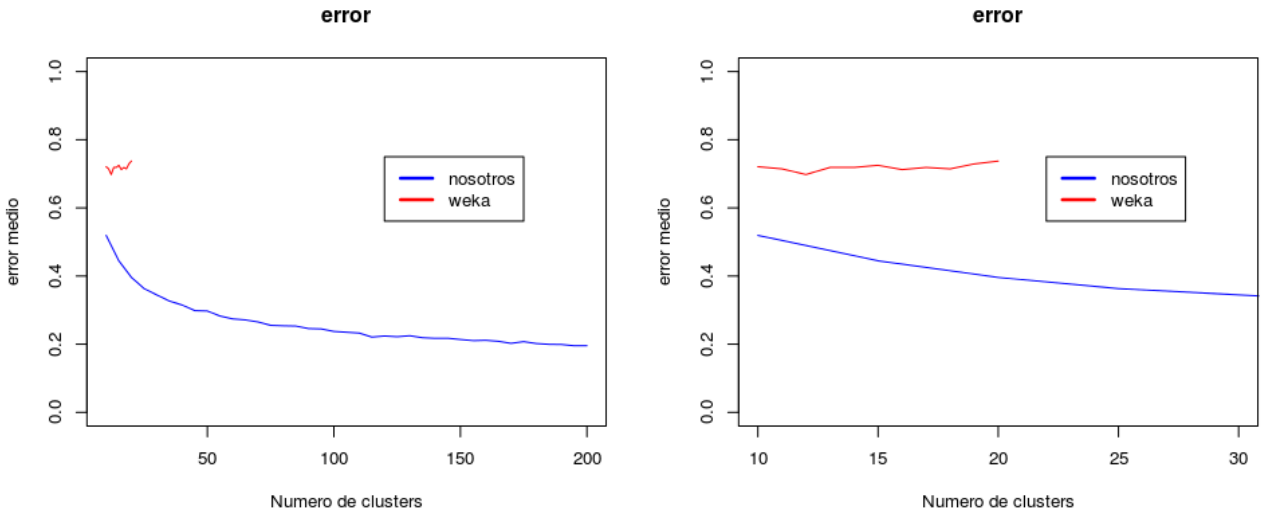


Apartado 3 – Evaluación extendida de K-means

El problema que podemos destacar ahora es el siguiente, como ya hemos comentado hay un mismo número que se puede escribir de formas distintas, por ejemplo si se revisa las imágenes usadas podemos ver que hay gente que ha escrito el 0 como un circulo y otra que lo ha escrito como un circulo al que le cruza una raya, evidentemente esto pude provocar que los haya varios grupos de números que representen una clase y sin embargo estén claramente separados en el espacio. Un método para resolver este problema es que a pesar de tener 10 clases usemos mas para englobar mejor estos números, es decir que varios clusters predigan la misma clase englobando patrones distintos.

Para ello hacemos la siguiente prueba, probamos nuestro algoritmo de 10 a 200 clusters con incrementos de 5 y nos devuelva el error medio de 40 pruebas. Por otro lado hacemos pruebas con weka usando de 10 a 20 clusters.

Obtenemos las siguientes gráficas, en una mostramos los datos obtenidos al usar de 10 a 200 clusters y en la otra los obtenidos entre 10 y 20.



Como se puede observar cuanto mas clusters usamos mejor predecimos hasta que llegamos a un punto donde la mejora no es significativa. El punto donde ya no mejorara sera cuando cada dato tenga asignado un cluster. Sin embargo no tiene sentido usar clustering si al final lo que vamos a hacer es tener un cluster por dato, ya que esto seria parecido a hacer vecinos próximos con k=1. Además computacionalmente es mas costos ya que el coste de test es el mismo que KNN ya que comprueba entre todos los

clusters cual es el mas cercano, y vecinos comprueba entre todos los vecinos cual es le mas cercano y además de ello tiene que hacer la parte de train, sobre un numero muy alto de clusters, que es cuando mayor coste tiene el algoritmo.

Solo hemos podido ejecutar en weka hasta k=20 porque el tiempo de ejecucion es muy elevado, además como se puede ver en la gráfica el resultado es independiente del numero de clusters. Creemos que esto puede ser por como funciona en weka, que asocia a cada cluster un valor, dejando los sobrantes sin clase.

Por ejemplo, esto es la salida de weka para k=14:

```
Cluster  0 <-- 5
Cluster  1 <-- 1
Cluster  2 <-- 6
Cluster  3 <-- 9
Cluster  4 <-- 0
Cluster  5 <-- No class
Cluster  6 <-- 3
Cluster  7 <-- No class
Cluster  8 <-- No class
Cluster  9 <-- 8
Cluster 10 <-- 2
Cluster 11 <-- No class
Cluster 12 <-- 4
Cluster 13 <-- 7
```

Apartado 4 – Evalución de Naive-Bayes y KNN sobre el conjunto de datos

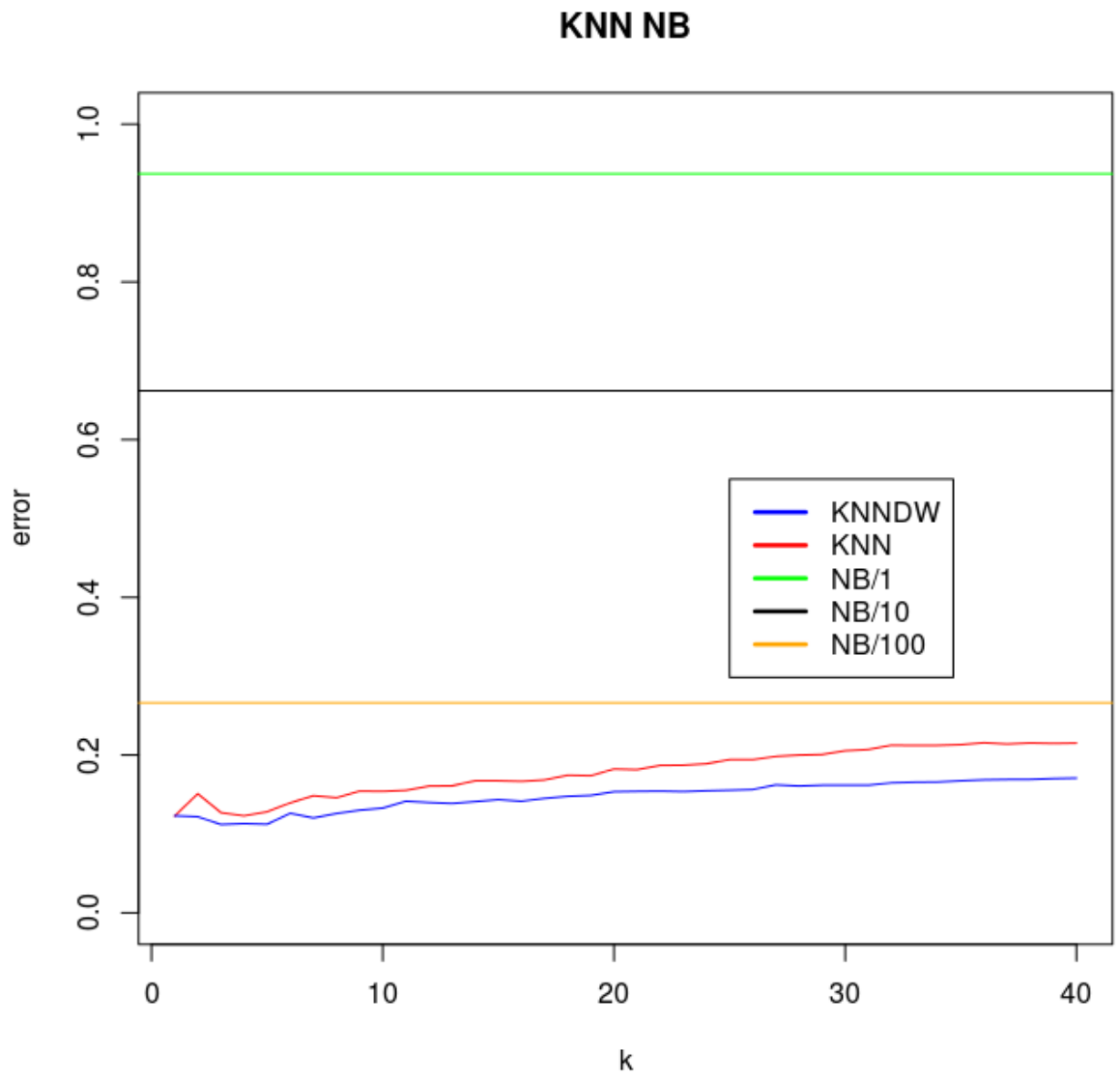
Clasificar los datos con KNN tiene mucho sentido ya que al recibir un nuevo dato para clasificar es muy probable que el dato mas cercano en distancia sea de la misma clase ya que si, por ejemplo, una persona escribe 2 veces el mismo número las similitudes entre uno y otro son muy significativas. Igualmente si 30 personas escriben el mismo número todos se parecerán mucho entre ellos, por tanto si clasificamos un numero del mismo tipo es muy probable que los vecinos mas cercanos sean ese grupo de 30 números.

El problema que presenta KNN es que el verdadero trabajo del algoritmo se encuentra en la fase de clasificación por tanto cada vez que queremos clasificar los datos tenemos que esperar una cantidad de tiempo que depende de la cantidad de vecinos que usemos y de la cantidad de datos que usemos, por tanto cuantos mas vecinos o mas datos tengamos mas se va a elevar el tiempo de espera de clasificación por cada dato a clasificar. Quizá ese se uno de los motivos por los que existe Kmeans, ya que ambos algoritmos son muy parecidos.

Clasificar los datos con Naive Bayes es bueno o no dependiendo de si redondeas los datos o no. Si no redondeas los datos y los tomas brutos se va a clasificar muy mal porque hay muy poca información, considerando el algoritmo muchas clases diferentes y con pocos datos en cada una. Si redondeas bien los datos se se van a quedar suficientes datos agrupados en una zona como para clasificar bien. Si no redondeas lo suficiente te vas a encontrar en un caso parecido a que no hayas redondeado.

Un pequeño inconveniente de NB es que es muy lento en entrenar. Dependiendo de la situación esto es preferible a que sea lento en clasificación o no, dependerá del contexto en el cual queremos clasificar los datos.

A continuación comparamos que combinaciones de algoritmos son mejores, se presentan aquí KNN, KNNDW que es como el anterior pero teniendo en cuenta la distancia, NB/1 que es Bayes normal, NB/10 que es Bayes redondeando las unidades de los datos y NB/100 que es Bayes redondeando las decenas.



Como se ve Naive Bayes es mejor redondeando cerca de las decenas, no hemos redondeado a las centenas porque es la escala máxima de los datos y ya no existiría información.

Sobre KNN comentar que no tiene sentido usar KNN con muchos vecinos porque clasifica casi igual o peor que con pocos, parece que el número adecuado de vecinos esta entorno a 4. También decir que es un poco mejor tener en cuanta la distancia hasta los vecino. Para comprobar esto hemos probado con numero mayores, como 150, obteniendo resultado peores que continúan con la tendencia del gráfico. Esto puede

deberse a que contamos con 48 ejemplo de cada numero (480/10), lo cual nos indica que mas de ese numero de vecinos es absurdo pues estaremos valorando mas vecinos de los que como máximo en común podemos tener, afectando a los resultados.

Por último hemos comparado estos resultados con los obtenidos por weka:

Hablando de NB hemos obtenido un error de 28,125%, similar a lo obtenido por nosotros cuando realizamos redondeo para agrupar los datos, por lo que podemos inferir que weka usa esta misma técnica.

Para el caso de vecinos próximos esto es lo que hemos obtenido:

K	KNN	KNNDW
1	21.25 %	21.25 %
4	23.3333 %	21.875 %
8	23.75 %	21.6667 %
12	25 %	23.3333 %
16	30.625 %	25.625 %
20	30.625 %	25.8333 %

Podemos observar como la tendencia es similar a la de nuestras implementaciones, aunque en la implementación de weka el punto de partida es algo mayor y además escala algo más rápido.

También concuerda que si incluimos la distancia en la votación de los vecinos el resultado mejora, como en nuestro caso.