ENTREGA PRÁCTICAS 2 PER

Elias Urios Alacreu

1.Ejercicios obligatorios 2
1.1 Multinomial 2
1.1 Gaussiana 3
2.Ejercicios opcionales 5
2.1. PCA+Clasificador Gaussiano 5
2.2. Bernoulli 6
2.3 KNN+L0, L1 y L3 7

1.Ejercicios obligatorios

1.1 Multinomial

La distribución multinomial presenta como motivación ser usada para datos que se tratan de contadores, es por eso que en una primera instancia podemos esperar que el uso aplicado a nuestro problema no sea del todo efectivo y además se requiere de un suavizado de Laplace al haber píxeles cuyo valor es 0. Teniendo esto en mente se ha diseñado una función (disponible en el fichero ***multinomial.m***) que dada una matriz de entrenamiento y otra a clasificar genere un clasificador multinomial. Como resulta obvio, debemos ir probando diferentes valores de *epsilon* para ver cuáles son los que nos proporciona el menor error. Para hacer esto se crea el script ***multinomial-exp.m*** y se le proporciona los valores de epsilon que se puede ver en el gráfico:

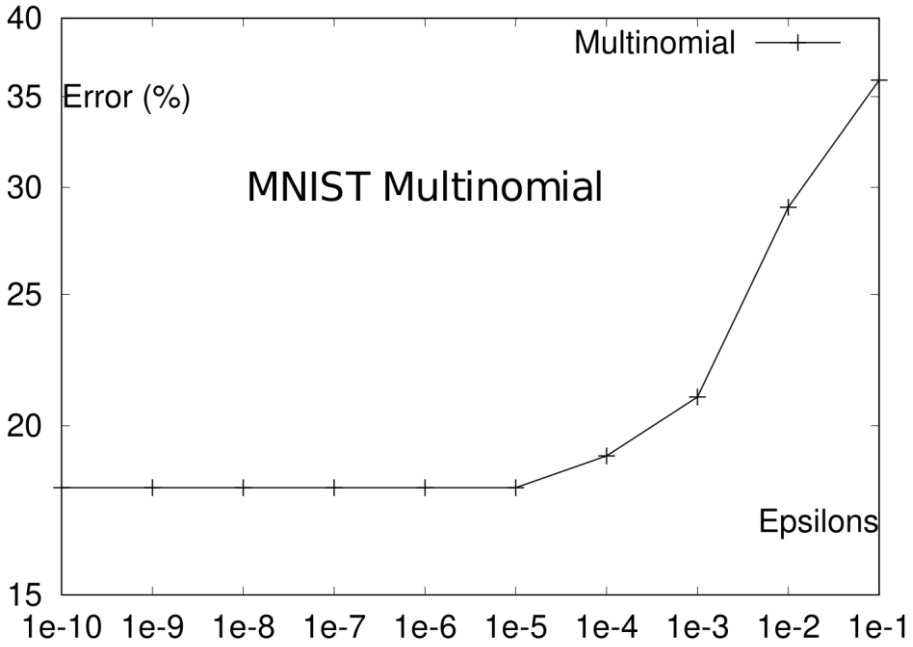

Figura 1. Tasa de error de la Multinomial en función de diversos valores de epsilon

Como se aprecia en la gráfica, el error se mantiene constante hasta llegar a 1e-4 donde aumenta considerablemente, es por eso que a la hora de llamar a ***multinomial-eva.m*** vamos a hacerlo con los valores hasta 1e-5. El resultado obtenido es el siguiente:

Figura 2. Error tras haber entrenado con diversos valores de Epsilon y con el conjunto de datos de prueba

La diferencia de error no resulta significativamente diferente, pero se puede apreciar que la que menor tasa de error ofrece es 1e-10 con un 16,32% de error. La mejor comparación que podemos obtener de nuestros resultados respecto a los que se nos presenta en la web del MNIST es con el clasificador lineal de una capa (perceptrón) que ofrece una tasa de error menor que la nuestra (12% vs 16,32%). Se trata de una conclusión lógica puesto que el diseño de nuestro clasificador lineal no está pensado para la tarea a tratar.

1.1 Gaussiana

La distribución Gaussiana puede resultar al estar pensada para vectores con datos de números reales, para llevar a cabo la implementación de esta distribución debemos tener en cuenta que las matrices de covarianzas calculadas para cada clase se tratan de

matrices singulares y debemos calcular la pseudoinversa en vez de la inversa y calcular el determinante a través de la suma de los valores propios. Además de esto, hay que aplicar un suavizado *flat smoothing* con diversos valores de k que se proporcionan. Todo el código se encuentra en el fichero ***gaussian.m***. Tras llamar al script ***gaussian-exp.m*** a ejecución, se obtiene la siguiente gráfica:

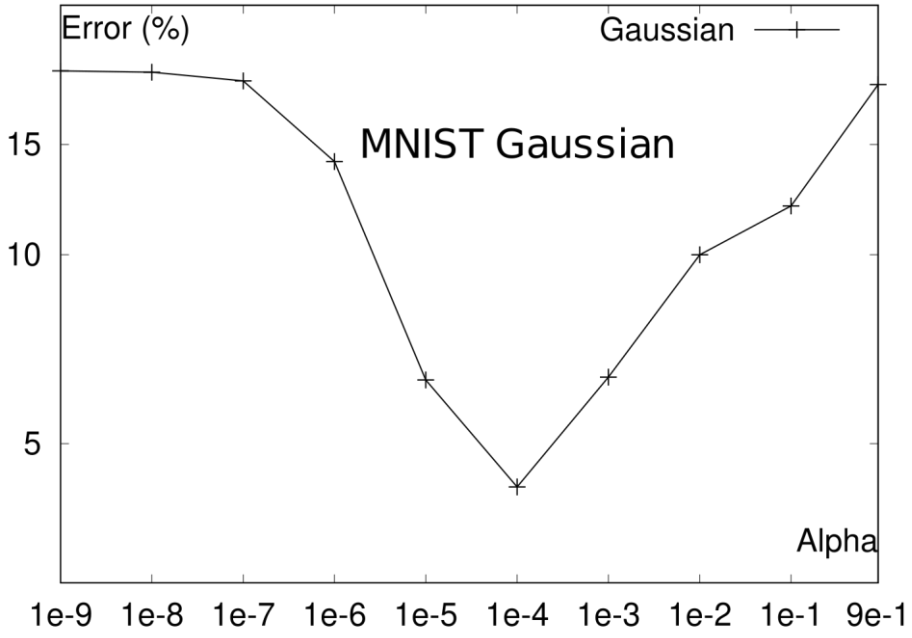

Figura 3. Tasa de error en función del valor del parámetro alfa en el clasificador Gaussiano

En base a los resultados obtenidos, resulta evidente que el mejor valor de Alpha se trata de 1e-4 y que proporciona un error de un 4,26%.

Si ahora llamamos al script ***gaussian-eva.m*** para entrenar el clasificador con este valor de Alpha propuesto, la tasa de error obtenida es:

```
(base) [elias@elias-ms7b86 Proyecto PER]$ ./gaussian-eva.m train-images-idx3-ubyte.mat.gz train-labels-idx1-ubyte.mat.gz t10k-images-idx3-ubyte.mat.gz t10k-labels-idx1-ubyte.mat.gz "[1e-4]"
ans = 0.041800
0.0001
4.18
```

Figura 4. Error obtenido con el conjunto de entrenamiento y el de prueba con el valor de Alpha propuesto

Vemos que se obtiene un error del 4,18%, si lo comparamos con los resultados que se presentan en la web del MNIST, concretamente con *40PCA+ quadratic classifier* vemos que hay una tasa de error de un 3,3%. Inferior a 1% por lo que podemos considerar el resultado del experimento satisfactorio

2.Ejercicios opcionales

2.1. PCA+Clasificador Gaussiano

Vamos a realizar la prueba con los valores de Alpha que menor tasa de error nos ha proporcionado el ejercicio anterior (1e-5,1e-4,1e-3) y vamos a hacer la proyección a K dimensiones (1,2,5,10,20,100,200,500). Ejecutamos ***pcag.m*** y observamos los resultados obtenidos:

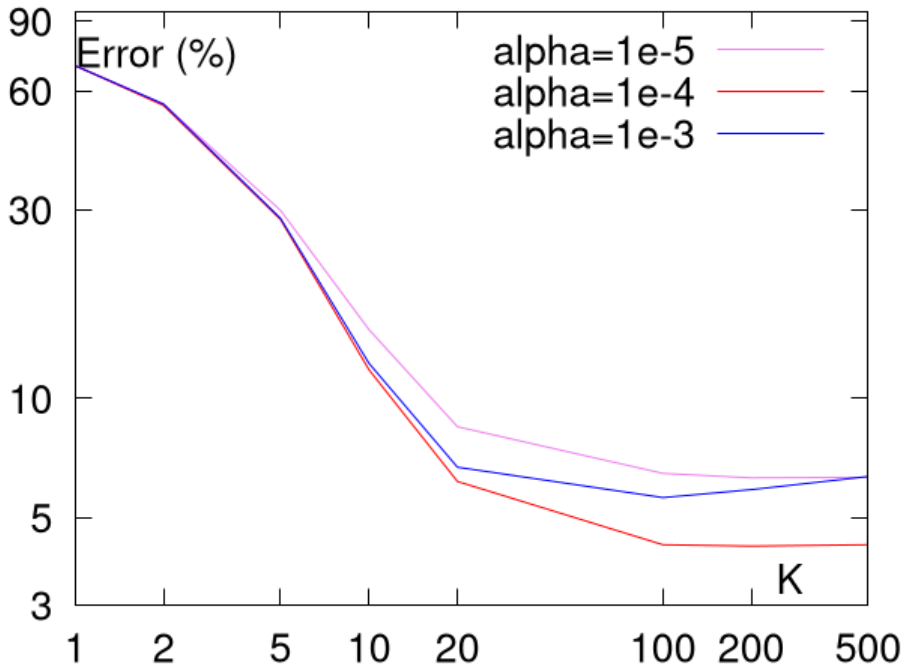

Figura 4. Tasa de error para diferentes valores de alfa y diversas dimensiones a proyectar

Al principio la tasa de error de los 3 valores de Alpha es muy parecida y el error es, lógicamente, muy alto, conforme nos vamos acercando a 100,200 y 500 dimensiones es donde se ven las diferencias notables, y observamos que el mejor valor es 1e-4 con K=200 (aunque no se pueda apreciar del todo bien en la gráfica, con K = 200 el error es del 4,23%

y con K=500 es del 4,26%). Si ahora hacemos los datos con el conjunto de entrenamiento y el de prueba (***pcageva.m***) se obtiene una tasa del error del 4,11%. Es decir, hemos mejorado nuestra tasa de error en 0.07% pero aun dista del error que se presenta en el MNIST del 3,3%. Aun así, los resultados del experimento deben ser considerados satisfactorios puesto que el tiempo de ejecución no se ve muy afectado y hemos sido capaces de mejorar la tasa de error.

```
(base) [elias@elias-ms7b86 Proyecto PER]$ ./pcageva.m train-images-idx3-ubyte.mat.gz train-labels-idx1-ub
yte.mat.gz t10k-images-idx3-ubyte.mat.gz t10k-labels-idx1-ubyte.mat.gz "[1e-4]" "[200]"
0.0001
4.11
```

Figura 5. Tasa de error con el valor de alfa y las k dimensiones propuestas

2.2. Bernoulli

La distribución de Bernoulli requiere llevar a cabo un proceso de normalización puesto que los datos se encuentran en una escala de grises (0...255), por lo que hay suministrar un umbral como parámetro para binarizar los datos. Por otro lado, es necesario, además trasladar un valor de epsilon para llevar a cabo un suavizado a través del truncamiento simple. Si tomamos como valores de epsilon 1e-20,1e-15,1e-10 y 1e-9 y los valores de umbral 0.1,0.2,0.4, 0.5 y 0.7. Se obtiene como resultado la siguiente gráfica:

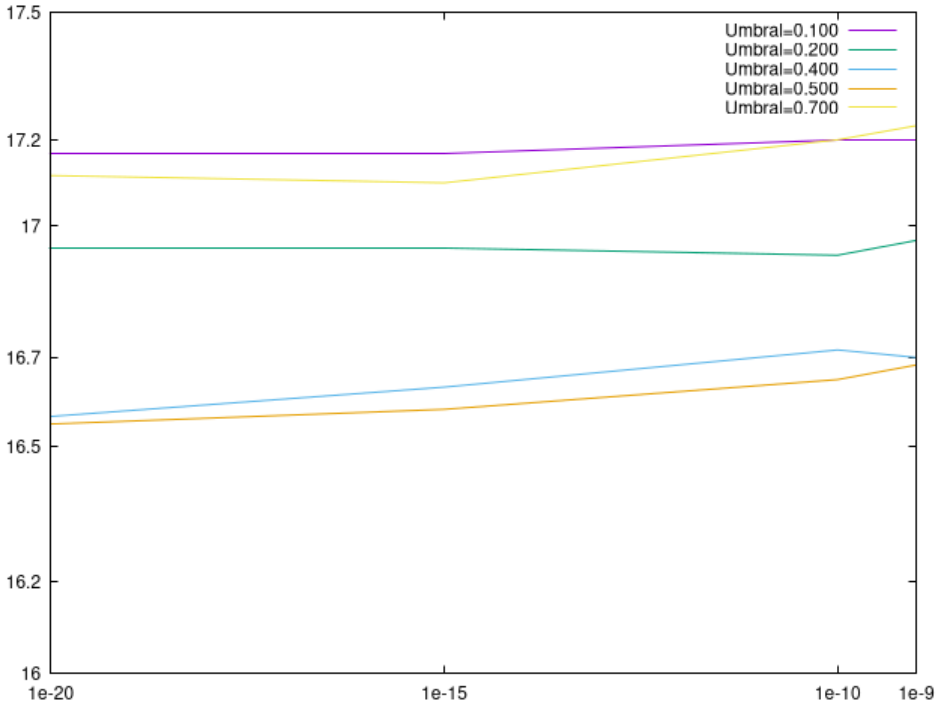

Figura 6. Tasa de error en función del umbral y el valor de epsilon

En base a los datos obtenidos, nos damos cuenta de que el mejor umbral y el mejor valor de epsilon es 0.5 y 1e-20, respectivamente. Si hacemos ahora la prueba con el conjunto de entrenamiento y el de datos con estos valores obtenemos la siguiente tasa de error:

```
(base) [elias@elias-ms7b86 Proyecto PER]$ ./bernoullieva.m train-images-idx3-ubyte.mat.gz train-labels-idx1-ubyte.mat.gz t10k-images-idx3-ubyte.mat.gz t10k-
labels-idx1-ubyte.mat.gz "[1e-20]" "[0.5]"
1e-20
15.62
```

Figura 7. Tasa de error con el umbral y epsilon propuestos para el conjunto de datos

Vemos que la tasa de error es del 15,62%, por lo que no es del todo buena si la comparamos con la gaussiano, pero es algo mejor que la multinomial

2.3 KNN+L0, L1 y L3

Para L3, sin aplicar PCA obtenemos un error de 2,65%, después de haber aplicado las diferentes proyecciones a k-dimensiones el gráfico resultante es el siguiente:

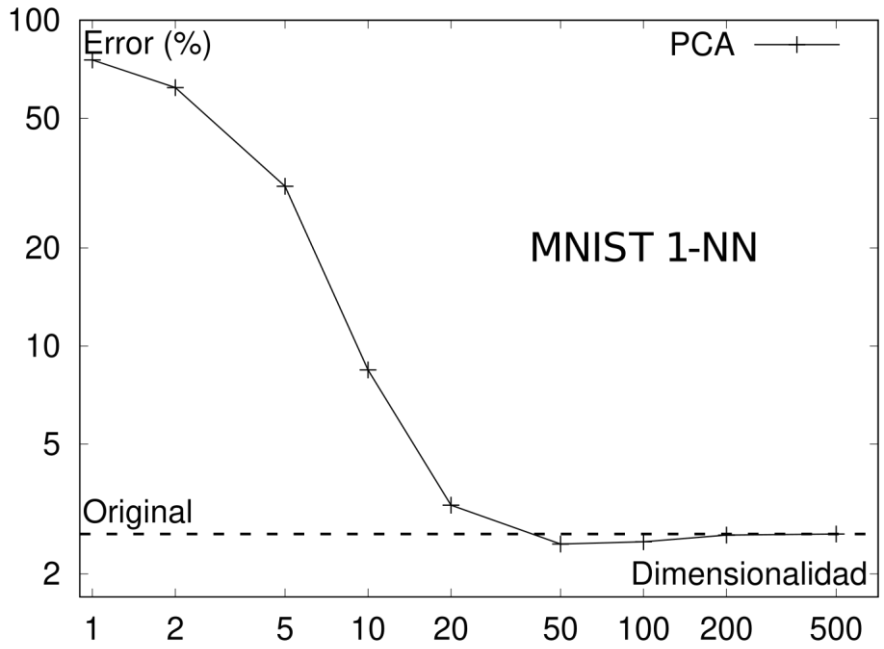

Figura 8. Tasa de error con diferentes dimensiones de PCA y distancia L3

Vemos que se alcanza un mínimo con k=50 donde el error ronda el 2,47% aproximadamente. Si ahora hacemos la evaluación con el conjunto de datos y el de entrenamiento, en L3 sin aplicar PCA el error resultante es del 2,83% (que coincide con los resultados presentados en la web del MNIST), y después de aplicar PCA con la k óptima mejoramos la tasa de error con un 2,71%, que resulta mejor que la tasa de error original, pero se queda algo por encima con respecto a otras técnicas que se aplican en la web del MNIST.

Vamos ahora con L1, la tasa de error sin aplicar PCA para este caso es del 3.51% aproximadamente, y si aplicamos PCA y usamos p=1 como distancia obtenemos los siguientes resultados:

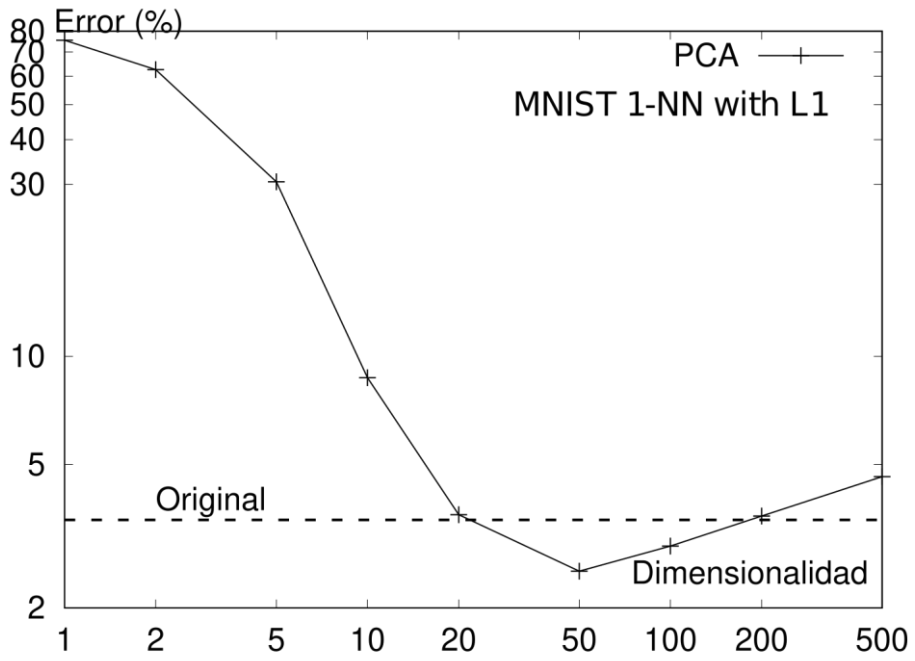

Figura 9. Tasa de error con L1 y diferentes dimensiones a proyectar

En este caso se vuelve a repetir que la mejor tasa de error se encuentra con k = 50, concretamente con una tasa de error del 2.53%.

Si hacemos ahora la evaluación obtenemos un 3.69% de error sin aplicar PCA y un 3.4% de error aplicando PCA con k = 50.

Finalmente, tenemos L0, para la cual la tasa de error es del 17% aproximadamente sin aplicar PCA, y, tras aplicar PCA obtenemos lo siguiente:

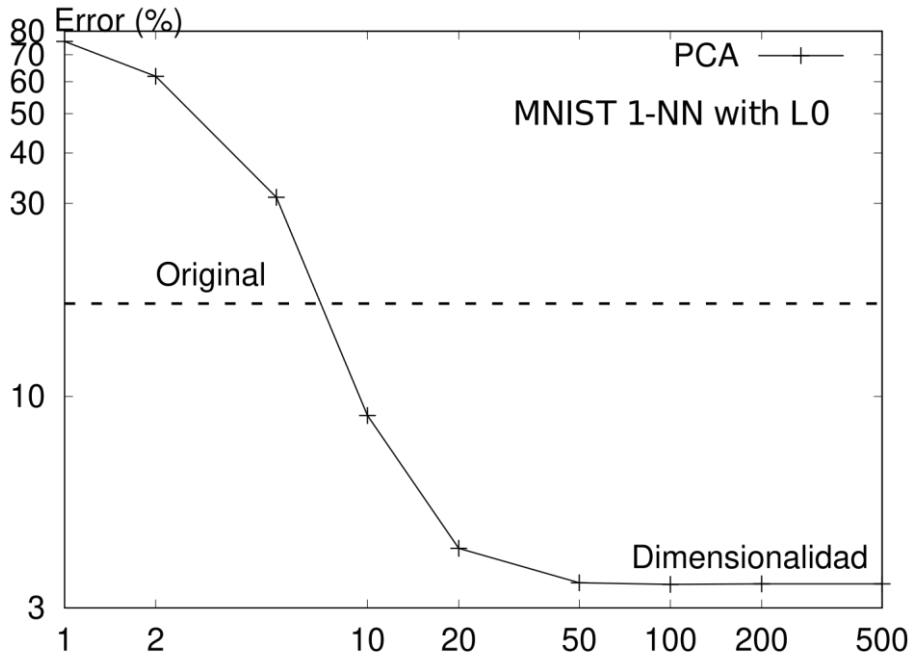

Figura 10. Tasa de error con L0 y diferentes dimensiones a proyectar

Se puede apreciar que a partir de k=100, la tasa de error se mantiene constante para todos los siguientes valores, así que se va a realizar la evaluación con k=100.

Pero antes observamos que el valor a la evaluación se trata de 17.42% (el más alto de todos) y que después de aplicar la técnica de PCA con la k propuesta obtenemos que la tasa de error es del 3.73%, una reducción bastante considerable